

175 PTAS
76
MI COMPUTER
CURSO PRACTICO DEL ORDENADOR PERSONAL,
EL MICRO Y EL MINIORDENADOR
sinclair
ZX Spectrum
Logo 1
sinclair
& Macmillan
Editorial Delta, S.A.

# MI COMPUTER CURSO PRACTICO
## DEL ORDENADOR PERSONAL, EL MICRO Y EL MINIORDENADOR

Publicado por Editorial Delta, S.A., Barcelona

Volumen VII-Fascículo 76

| | |
|---|---|
| Director: | José Mas Godayol |
| Director editorial: | Gerardo Romero |
| Jefe de redacción: | Pablo Parra |
| Coordinación editorial: | Jaime Mardones<br>Francisco Martín |
| Asesor técnico: | Ramón Cervelló |

Redactores y colaboradores: G. Jefferson, R. Ford, F. Martín, S. Tarditti, A. Cuevas, F. Blasco
Para la edición inglesa: R. Pawson (editor), D. Tebbutt (consultant editor), C. Cooper (executive editor), D. Whelan (art editor), Bunch Partworks Ltd. (proyecto y realización)

Realización gráfica: Luis F. Balaguer

Redacción y administración:
Paseo de Gracia, 88, 5.°, 08008 Barcelona
Tels. (93) 215 10 32 / (93) 215 10 50 - Télex 97848 EDLTE

MI COMPUTER, *Curso práctico del ordenador personal, el micro y el miniordenador*, se publica en forma de 96 fascículos de aparición semanal, encuadernables en ocho volúmenes. Cada fascículo consta de 20 páginas interiores y sus correspondientes cubiertas. Con el fascículo que completa cada uno de los volúmenes, se ponen a la venta las tapas para su encuadernación.
El editor se reserva el derecho de modificar el precio de venta del fascículo en el transcurso de la obra, si las circunstancias del mercado así lo exigieran.


ISBN: 84-85822-83-8 (fascículo) 84-7598-067-2 (tomo 7)
84-85822-82-X (obra completa)
Depósito Legal: B. 52-84

Fotocomposición: Tecfa, S.A., Pedro IV, 160, Barcelona-5
Impresión: Cayfosa, Santa Perpètua de Mogoda (Barcelona) 268506
Impreso en España-Printed in Spain-Junio 1985

Editorial Delta, S.A., garantiza la publicación de todos los fascículos que componen esta obra.

Distribuye para España: Marco Ibérica, Distribución de Ediciones, S.A., Carretera de Irún, km 13,350. Variante de Fuencarral, 28034 Madrid.
Distribuye para Colombia: Distribuidoras Unidas, Ltda., Transversal 93, n.° 52-03, Bogotá D.E.
Distribuye para México: Distribuidora Intermex, S.A., Lucio blanco, n.° 435, Col. San Juan Tlihuaca, Azcapotzalco, 02400, México D.F.
Distribuye para Venezuela: Distribuidora Continental, S.A., Edificio Bloque Dearmas, final Avda. San Martín con final Avda. La Paz, Caracas 1010.

Pida a su proveedor habitual que le reserve un ejemplar de MI COMPUTER. Comprando su fascículo todas las semanas y en el mismo quiosco o librería, Vd. conseguirá un servicio más rápido, pues nos permite realizar la distribución a los puntos de venta con la mayor precisión.

**Servicio de suscripciones y atrasados** (sólo para España)

Las condiciones de suscripción a la obra completa (96 fascículos más las tapas, guardas y transferibles para la confección de los 8 volúmenes) son las siguientes:

a) Un pago único anticipado de 19 425 ptas. o bien 8 pagos trimestrales anticipados y consecutivos de 2 429 ptas. (sin gastos de envío).

b) Los pagos pueden hacerse efectivos mediante ingreso en la cuenta 6.850.277 de la Caja Postal de Ahorros y remitiendo a continuación el resguardo o su fotocopia a Editorial Delta, S.A. (Paseo de Gracia, 88, 5.°, 08008 Barcelona), o también con talón bancario remitido a la misma dirección.

c) Se realizará un envío cada 12 semanas, compuesto de 12 fascículos y las tapas para encuadernarlos.

Los fascículos atrasados pueden adquirirse en el quiosco o librería habitual. También pueden recibirse por correo, con incremento del coste de envío, haciendo llegar su importe a Editorial Delta, S.A., en la forma establecida en el apartado b).

Para cualquier aclaración, telefonear al (93) 215 75 21.

**No se efectúan envíos contra reembolso.**



Un día un perito
fue a un bosque y
vio una abeja y
un poco de miel. A
él le encantava la
miel. dijo me encanta
la miel y voi a cojerla,
así que trepó y metió la
nariz en un agujero. una
abeja lo picó en la
nariz. se fue corriendo a
su casa y nunca más
volvió a ir al bosque. fin

Un día un perito fue a un bosque y vio una abeja y un poco de miel A el le encantava la miel. dijo me encanta la miel y voy a cogerla asi que trepó y metió la nariz en un agujero una abeja grande lo pico en la nariz se fue corriendo a su casa y nuunca mas volvio a ir al bosque Fin.

# Ayuda didáctica

**La utilización de micros puede ayudar eficazmente a desarrollar las aptitudes para la escritura y el dibujo en los niños**

Por regla general, las innovaciones técnicas suscitan una respuesta cautelosa por parte del público: la gente suele mostrarse reacia a la experimentación y se niega a apoyar nuevas ideas hasta no haberlas probado por sí mismas. Una de las herramientas básicas de la educación en la actualidad, el cuaderno de ejercicios, representó un desarrollo revolucionario cuando empezó a reemplazar a la pizarra. Previamente, el papel se había venido utilizando mayormente para dejar constancia de transacciones comerciales y para los trabajos importantes de las universidades, el gobierno y la Iglesia. Comodidad cara, se consideró bastante inadecuada para los niños, quienes probablemente no sabrían cómo utilizarlo o lo desaprovecharían.

Sin embargo, la introducción de los cuadernos de ejercicios les proporcionó a los estudiantes un registro permanente de su trabajo. Les permitió dividir su atención entre diferentes tareas escritas sin tener para ello que borrar sus esfuerzos anteriores. A pesar de que fue una innovación cara, fue inmensamente valiosa. No obstante, por cuanto concierne a los argumentos de la economía, el empleo de los micros despierta objeciones similares a las que generaron en su día los cuadernos de ejercicios. El concepto de un ordenador por niño es inaceptable para muchas personas sólo desde el punto de vista económico y, lamentablemente, éste ha sido siempre un factor preponderante para la determinación de la política educativa.

Por cuanto respecta a los micros existe, asimismo, la dificultad de persuadir a los maestros de que acepten una tecnología nueva que ya ha adquirido una reputación dudosa, principalmente a consecuencia de la herencia de las técnicas de la "enseñanza programada" que vimos en el capítulo anterior. Una mirada atenta a la forma en que los ordenadores se pueden utilizar en la clase despejará muchas objeciones y demostrará que el ordenador no sólo aumenta la eficacia (resultando más económico de lo que sugeriría su costo inicial), sino que también puede suponer una valiosa contribución a los procesos creativos del aprendizaje.

## Escritura creativa

Consideremos primero el uso de los micros para ayudar a desarrollar aptitudes para la escritura. Escribir una historia implica varios procesos creativos diferentes. Tras la conceptualización, antes de completar la historia es necesario reescribirla, editarla y mejorarla. En este sentido, se considera que escribir es muy similar a, supongamos, pintar y esculpir.

Lamentablemente, la forma en que se espera que muchos niños produzcan trabajo escrito en las

**Juego de palabras**
Uno de los problemas principales con los que se encuentra el niño cuando intenta escribir de forma creativa es la incapacidad de sus aptitudes de escritura manual para marchar al mismo paso que sus pensamientos. En el proceso de formar las letras en la forma correcta, es muy fácil perder el hilo del pensamiento. Por este motivo es característico que la redacción del niño parezca desarticulada. El procesador de textos no soluciona este problema; hallar una tecla puede ocupar tanto tiempo como escribir la misma letra a mano. Pero con unas adecuadas aptitudes de mecanografía, que exigen menos aptitudes motoras de precisión, el niño puede en cierta medida superar esta dificultad. Puesto que se puede efectuar la edición ya sea en pantalla o en papel antes de contar con un producto ya acabado, se puede invertir menos tiempo en hacer que la historia aparezca en el papel, por lo cual los pensamientos y las palabras se articulan en un grado superior

**Micro-cognición**
El empleo de un ordenador para tratamiento de textos y dibujo desarrolla muchas de las mismas aptitudes que la escritura y el dibujo manuales. Pero el ordenador, utilizado adecuadamente, en realidad puede enseñar o reafirmar muchas aptitudes mejor que si se las aprendiera a mano, y desarrolla asimismo otras adicionales. En las tablas se relacionan algunas de las aptitudes cognoscitivas que los ordenadores pueden enseñar o perfeccionar

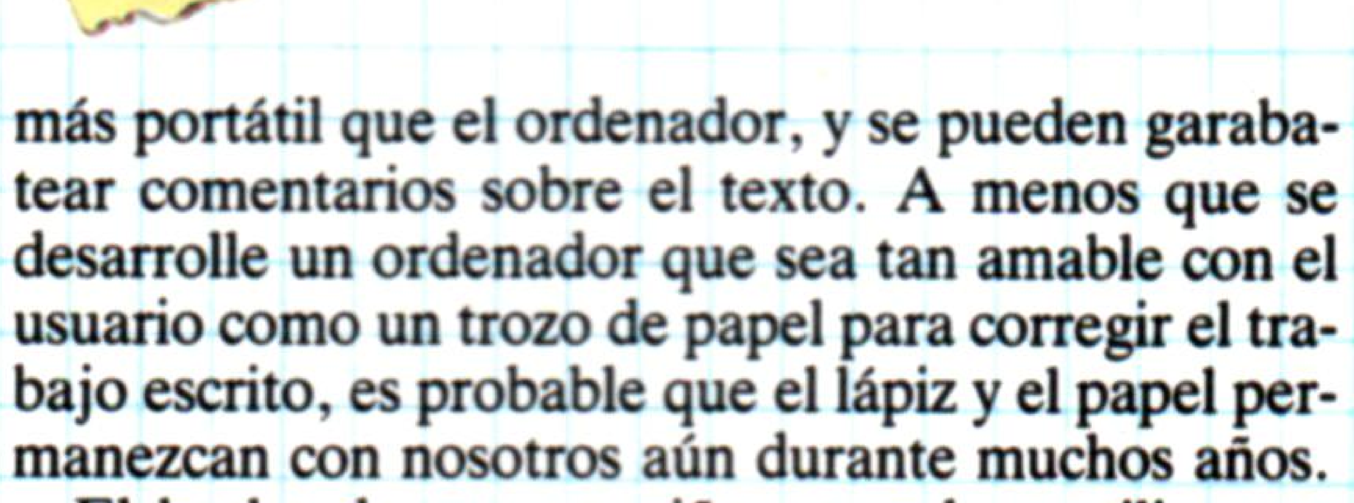
## Aptitudes cognoscitivas del tratamiento de textos

- Aptitudes motoras precisas (pequeños movimientos de dedos y manos)
- Reconocimiento de letras
- Formación de letras (un juego de caracteres apropiado en la pantalla y la impresora pueden enseñar la forma correcta de dibujar una letra)
- Corrección de pruebas y revisión de errores
- Ortografía (múltiples aptitudes específicas)
- Secuenciación (colocación de los eventos en el orden correcto)
- Memoria visual (aprender la posición de las teclas de modo que no sea necesario mirarlas)



## Aptitudes de los gráficos

- Coordinación ojo-mano
- Aptitudes motoras de precisión
- Diseño
- Discriminación de formas
- Relación causa-efecto
- Fluidez (generación de muchísimas ideas con mucha rapidez)
- Flexibilidad (pensamiento lateral: ver lo antiguo bajo una nueva luz)
- Imaginación



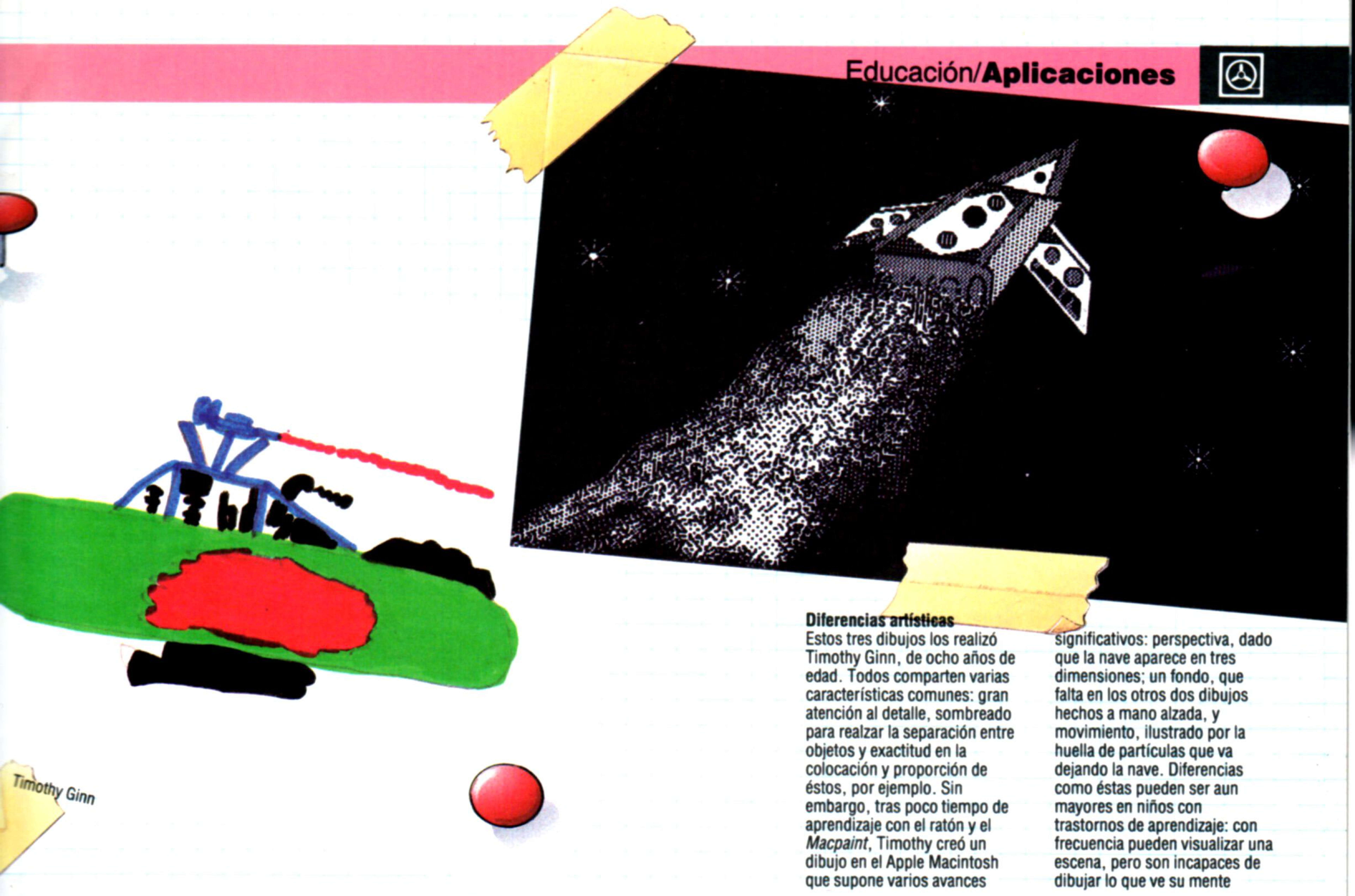
escuelas se opone al proceso creativo. Con frecuencia, la exigencia primordial del maestro es que acaben el trabajo lo más rápidamente posible. También han de presentarlo pulcro, sin tachaduras ni correcciones. El deseo del maestro de obtener un trabajo limpio suele ser perjudicial para la edición, la nueva redacción y, por tanto, la escritura creativa seria. La actitud del maestro está determinada en gran medida por la necesidad de procesar el trabajo escrito con la mayor rapidez posible, y corregir los trabajos sucios lleva más tiempo.

Permitir que un niño utilice un ordenador como procesador de textos reconcilia el síndrome "de página pulcra" con las aptitudes para la escritura seria. Los niños aprenden a editar, corregir y mejorar su trabajo y aun así terminan con una copia final pulcramente impresa. El texto se puede almacenar en disco y cuando la tarea está acabada el niño puede entregarle el disco al maestro, quien lo podrá colocar en un ordenador en su casa o en la escuela y revisarlo cuando sea conveniente. Si se utiliza una red de ordenadores, el maestro puede llamar directamente al archivo del niño y controlar su trabajo. El producto final es una copia pulcra sobre papel de la que hasta el escritor más descuidado se sentirá orgulloso. Existen, asimismo, programas que revisan un archivo de texto para comprobar la ortografía, la puntuación y corregir cualquier eventual error.

Un posible inconveniente de este proceso es que el niño, al ver el texto corregido, no adquiriera conciencia de sus errores como ocurriría en el caso de un cuaderno de ejercicios corregido. Muchas personas que utilizan procesadores de textos prefieren corregir su manuscritos sobre el papel en vez de en la pantalla. Es más fácil para los ojos, el papel es más portátil que el ordenador, y se pueden garabatear comentarios sobre el texto. A menos que se desarrolle un ordenador que sea tan amable con el usuario como un trozo de papel para corregir el trabajo escrito, es probable que el lápiz y el papel permanezcan con nosotros aún durante muchos años.

El hecho de que un niño aprenda a utilizar un ordenador ocasiona muchos comentarios sobre la "barrera del teclado", pero se habla muy poco de la "barrera del lápiz" cuando se aprende a escribir a mano. Uno de los principales objetivos de los maestros de preescolar o de primaria es el de enseñar al niño a escribir. Como primer paso éste debe aprender a sostener el lápiz correctamente, luego aprender las formas de las letras y, por último, reproducir esas formas sobre el papel. Las letras se deben dibujar de una manera determinada y las partes de las letras en un orden determinado. El niño debe hacer palabras de un tamaño uniforme y asegurarse de trazarlas a lo largo de una línea recta. Dominar estas destrezas requiere años de arduo trabajo y muchos niños jamás consiguen escribir adecuadamente. Como si esto no fuera suficiente, el niño de ocho o nueve años debe volver a aprender a escribir con letras cursivas o "unidas".

Compare este proceso con aprender mecanografía al tacto. Uno debe aprender a reconocer las diferentes formas de letras y sus posiciones en el teclado, y a alcanzar ciertas letras con dedos específicos. Esto es mucho más fácil, aunque algunas personas podrían aducir que el desafío que supone aprender a escribir es de gran valor y no se puede ignorar. Pero dominar la escritura a mano a menudo es una barrera para la expresión escrita. Si a un grupo de niños se les asignara cierto límite de tiempo para producir un trabajo, la producción de cada




**Diferencias artísticas**
Estos tres dibujos los realizó Timothy Ginn, de ocho años de edad. Todos comparten varias características comunes: gran atención al detalle, sombreado para realzar la separación entre objetos y exactitud en la colocación y proporción de éstos, por ejemplo. Sin embargo, tras poco tiempo de aprendizaje con el ratón y el *Macpaint*, Timothy creó un dibujo en el Apple Macintosh que supone varios avances significativos: perspectiva, dado que la nave aparece en tres dimensiones; un fondo, que falta en los otros dos dibujos hechos a mano alzada, y movimiento, ilustrado por la huella de partículas que va dejando la nave. Diferencias como éstas pueden ser aun mayores en niños con trastornos de aprendizaje: con frecuencia pueden visualizar una escena, pero son incapaces de dibujar lo que ve su mente

niño dependería de sus aptitudes para escribir a mano. Sólo cabría esperar una línea o dos de un niño de cinco años de edad, tres o cuatro líneas de uno de seis y quizá una página de un niño de siete años. Si se les enseñara a los niños a mecanografiar, los maestros (y los padres) se sorprenderían de la calidad y cantidad de sus resultados.

El procesador de textos, por supuesto, no ha hecho que la escritura manual quede obsoleta, pero en el transcurso de la vida la práctica común dicta cada vez más la utilización de un teclado, tanto para los negocios como para las situaciones sociales. La habilidad con el teclado a menudo sólo se les enseña a los jóvenes en cursos comerciales y la extensión de estas aptitudes a los niños en edad escolar es beneficiosa. Esta tendencia continuará en la medida en que la presencia de ordenadores se vuelva más familiar y su empleo más diversificado.

## Trabajo de diseño

Otro aspecto de la microinformática que reviste gran importancia para el niño es la creciente facilidad con que se pueden crear sofisticados gráficos gracias a la utilización de un ordenador. Este papel del micro se ha visto seriamente limitado por su precio: el proceso de gráficos por lo general requiere un hardware más complejo y más caro que el proceso de textos. Sin embargo, esta característica de la informática escolar está evolucionando, puesto que simultáneamente las máquinas se están volviendo más potentes y menos costosas.

Al principio, la tablilla para gráficos permitía utilizar en la pantalla varias funciones artísticas, pero recientemente estas características se han incorporado en forma de "tecnología de ratón", entre otras máquinas, en el Apple Macintosh. Al permitir la selección de "iconos", el ratón le ofrece al usuario un control preciso e inmediato sobre una amplia gama de actividades, desde dibujar imágenes y formas a reducir y ampliar imágenes, rellenando superficies con patrones y manipulando la visualización de diversas maneras.

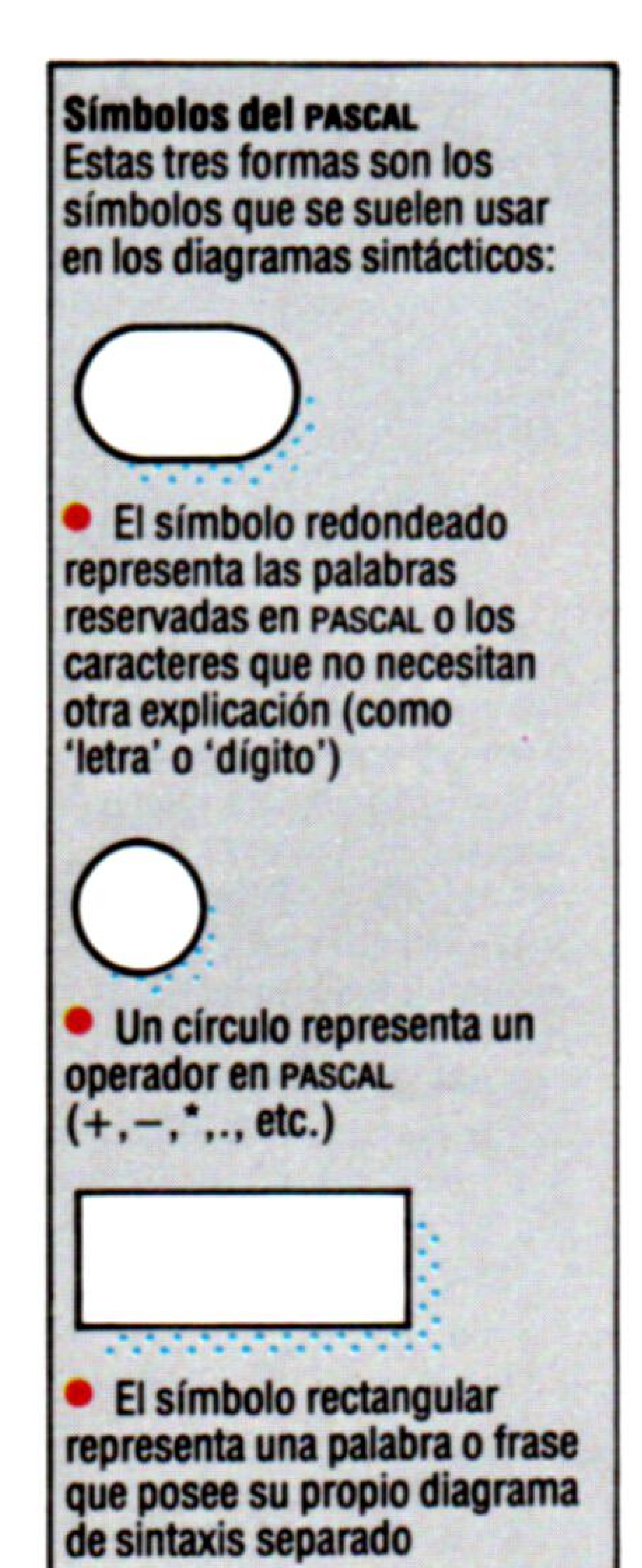
Además, el ordenador le permite al estudiante integrar y relacionar material diferente, textual y gráfico, con más facilidad que los métodos tradicionales. Una imagen acabada se puede "recortar" y "pegar" en un fragmento de texto generado anteriormente. Aunque los programas para gráficos tales como el *Macpaint* no pueden sustituir a las prácticas artísticas que se realizan en la clase, sí le ofrecen al niño un medio con nuevas técnicas con las que experimentar, aprender y expresarse.

El niño desea aprender. Su curiosidad innata lo obliga a pasarse cada una de las horas de vigilia buscando, explorando y aprendiendo todo lo que puede acerca de sus nuevos entornos, y expresando sus impresiones a través de un impulso creador. Los lápices de colores se cogen con ansiedad, dibujando sobre el papel o la pared más cercanos: así se van desarrollando estas aptitudes básicas.

Los niños de dos años con acceso a un ordenador efectúan sus investigaciones con el mismo regocijo, pulsando todas las teclas y viendo aparecer en la pantalla pequeñas formas. Aceptan la nueva tecnología con mayor facilidad que los adultos y, al encontrarse con un entorno que contiene ordenadores, el niño los utiliza más fácil y naturalmente. Ahora ya sólo es cuestión de tiempo para que las escuelas incorporen el micro como una herramienta educativa vital y dejen de considerarlo sólo como un juguete muy caro.

# Tipos de variables

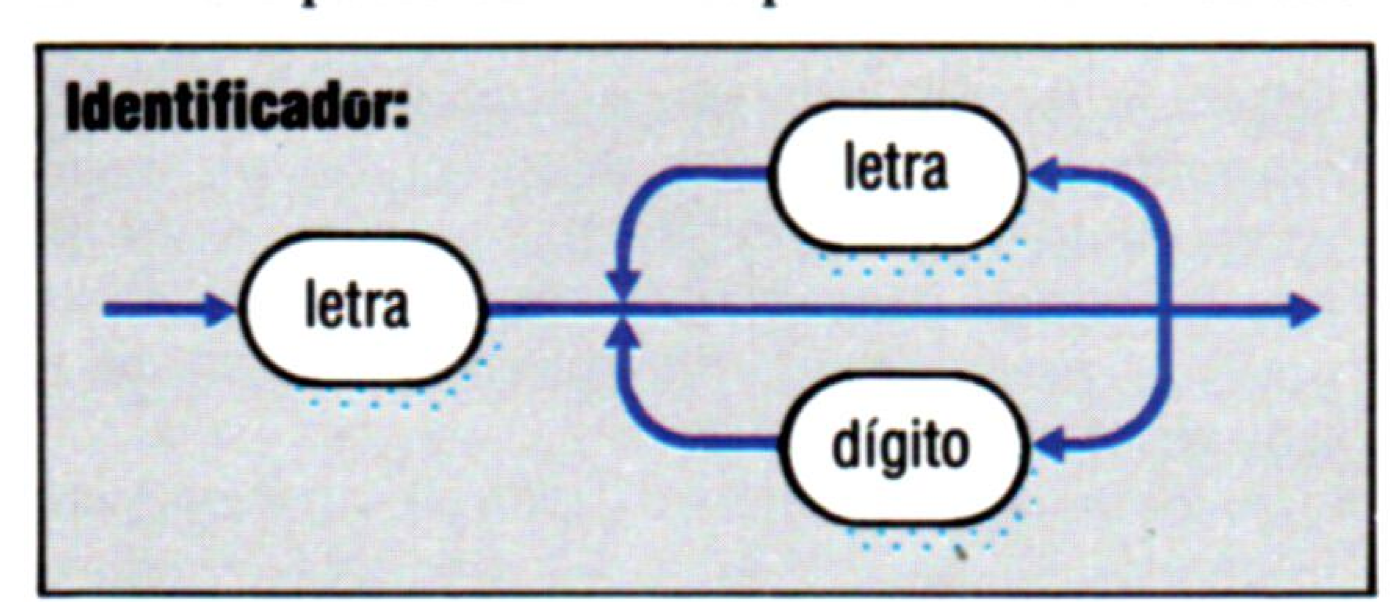
## En este capítulo estableceremos las diferencias entre las variables y estudiaremos las sentencias compuestas


**Símbolos del PASCAL**
Estas tres formas son los símbolos que se suelen usar en los diagramas sintácticos:

- El símbolo redondeado representa las palabras reservadas en PASCAL o los caracteres que no necesitan otra explicación (como 'letra' o 'dígito')
- Un círculo representa un operador en PASCAL (+,−,*,.., etc.)
- El símbolo rectangular representa una palabra o frase que posee su propio diagrama de sintaxis separado

El PASCAL proporciona cuatro tipos de datos simples ya predefinidos para nosotros, y a los que se les asignan los identificadores Integer (enteros), Real (reales), Char (caracteres) y Boolean (booleanos). Los números se clasifican según posean una parte de fracción (números reales) o sean números enteros naturales (enteros). Por supuesto, la verdadera gama de números disponibles está determinada por la cantidad de bytes que se empleen para almacenar un valor dado de cada tipo. Las decisiones de diseño como ésta las toma el escritor del compilador (el *implementador*) y se dice que tales características están *subordinadas a la implementación*. Todas estas subordinaciones a la implementación deben especificarse en la documentación para que un compilador sea autorizado por la normativa de la ISO (Internacional Standards Organization).

Es casi seguro que la gama de enteros de su compilador esté comprendida entre −32.768 y 32.767, o bien entre −2.147.483.648 y 2.147.483.647, según se utilice una representación de dos o cuatro bytes. El PASCAL posee un nombre para el valor del entero




máximo, representado mediante el identificador de constante predefinida, MaxInt. Por tanto, puede hallar fácilmente este valor mediante la sentencia:

```
WriteLn ('MaxInt is :',MaxInt)
```

Los números reales también se mantienen en una gama y exactitud limitadas: por lo general desde alrededor de 1.7E+38, con una precisión de seis o siete dígitos en el peor de los casos. Esta forma de escribir los números reales, llamada *notación científica*, es la forma por defecto del PASCAL, pero se puede utilizar la manera más normal de escribirlos, con el punto decimal (p. ej., 123.456), si así lo cree conveniente.

Se ha de tener en cuenta, no obstante, que un número real *siempre* posee un punto decimal separando la parte entera de la parte fraccionaria, y que *ambas deben* estar presentes. De modo que 0.1 y 1.0E-1 son aceptables, pero .1 o 1E-1 son ilegales. Afortunadamente, estas reglas estrictas sólo se aplican a números que en el texto del programa se deban reconocer como reales. Si usted está entrando datos desde el teclado, por ejemplo, un entero se leerá y se convertirá automáticamente si se espera un valor real.

Char es la abreviatura de carácter, por supuesto, y un valor de este tipo será uno de los miembros del juego de caracteres (por lo general ASCII) de que disponga el ordenador. El PASCAL asegura su propia portabilidad dando por sentado que:

- Los caracteres de la A a la Z están ordenados alfabéticamente, lo que significa que, según el valor de los caracteres, A es menor que B, B menor que C, y así sucesivamente.
- Los caracteres de números del 0 al 9 están ordenados y contiguos, lo que significa que, sea cual fuere el valor de 0, 1 será el siguiente, etc.

El juego de caracteres ASCII también posee un alfabeto contiguo, pero ello no es esencial para el PASCAL, ¡sino para los programadores de este lenguaje! Cada valor de carácter tendrá un código numérico, que es un valor de un subrango del tipo de enteros. Los códigos ASCII están definidos en la gama entre 0 y 127, y muchas máquinas lo amplían hasta 255 para códigos extras de caracteres para gráficos. Podemos fácilmente hacer un mapa de cualquier juego de caracteres en la escala de los valores "comunes" que utiliza internamente el ordenador. El PASCAL proporciona la función predefinida Ord: ésta devuelve el código de entero de su argumento; de modo que Ord (A) es el equivalente de 65 en el juego de caracteres ASCII. Otra función, chr, proporciona la función inversa: chr (65) da el carácter A (observe que con chr *no* se utiliza el signo dólar).

Tanto el rango de los valores de caracteres como los enteros se definen para cada implementación, y existen en una escala ordenada de constantes conocidas. Por este motivo se los denomina tipos *ordinales* o *escalares*. Cualquiera que sea el valor, siempre sabemos cuáles son los anteriores y los siguientes, si es que los hay. Estos valores adyacentes se pueden obtener mediante las dos funciones escalares:

```
pred(elemento) (predecesor)
succ(elemento) (sucesor)
```

Por consiguiente, succ (3) dará siempre el valor de carácter 4, pero pred(Z) sólo será Y en algunos juegos de caracteres, como el ASCII. Pred (MaxInt) será ya sea 32.766 o bien 2.147.483.646. La función chr sólo se puede utilizar con un argumento que sea un código de carácter. Todas las otras funciones escalares se pueden emplear con cualquier tipo escalar, aunque si se utiliza ord con enteros, devuelve el valor de su argumento.

Las variables booleanas son el más simple de todos los tipos escalares, porque en la escala hay sólo dos valores: false (falso) y true (verdadero), por ese orden. Puesto que son tipos escalares simples, las funciones escalares se pueden aplicar a cualquier valor booleano: el valor ordinal de false es 0 y ord (true) es 1. Las otras funciones escalares, pred y succ, sin embargo, no son de gran utilidad aquí. Su



compilador de PASCAL desaprobará categóricamente que usted pruebe algo como WriteLn(pred(false)): no debe sorprenderle que sea un error intentar evaluar un valor inexistente.

La siguiente parte de definición de constantes de un programa muestra todos los tipos ordinales simples tal como podrían aparecer en un texto fuente.

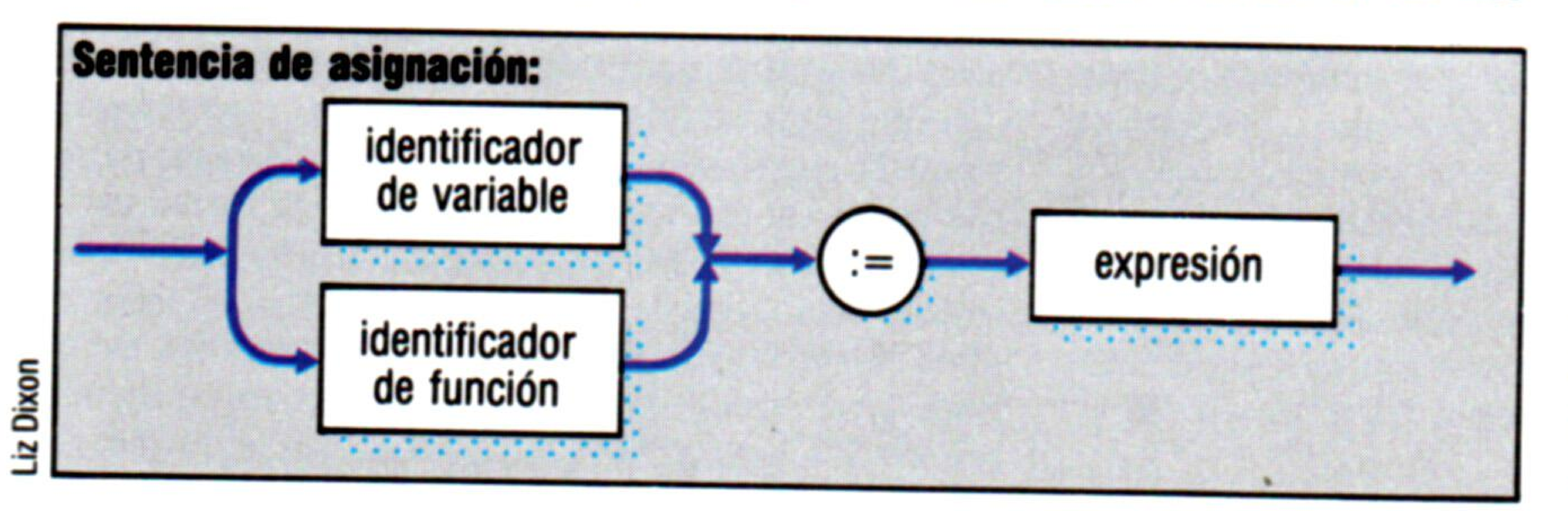
```
CONST
  VAT          = 0.15;
  columnas     = 40;
  espacio      = ' ';
  depuracion   = false;
```

## Igualdad y asignación

El signo de igualdad (=) *siempre* significa es igual a en PASCAL, y se utiliza para igualar identificadores de constantes a los valores que retienen. Cuando declaramos variables en la sección de declaraciones VAR, los dos puntos (:) separan al identificador de variables recién definido de su tipo. Por ejemplo:

```
VAR
  coeficiente   : real;
  numero        : integer;
  simbolo       : char;
  hecho         : boolean;
```

Cuando queremos asignar valores a estas variables, se utiliza el *operador de asignación* compuesto (:=). Éste ayuda a diferenciar claramente las tres clases de operación. Las definiciones de CONST igualan valores permanentes, las declaraciones VAR sólo reservan espacio de memoria, y asignación le da al identificador un valor (tal vez temporal).




## La sentencia compuesta

Cuando se deben ejecutar dos o más sentencias como parte de un único proceso, podemos encerrarlas entre paréntesis entre las palabras BEGIN y END como una sentencia "compuesta". Recuerde que cada sentencia integrante debe estar separada de cualquier sentencia que le siga mediante un punto y coma. Ya hemos visto sentencias compuestas, dado que el cuerpo de sentencias ejecutables de todos los programas asume esta forma. A continuación ofrecemos un programa completo que utiliza muchas de las características que ya hemos considerado. Adoptaremos la convención de escribir las palabras reservadas en mayúsculas para distinguirlas de los identificadores. Observe las líneas dejadas en blanco entre cada parte de la estructura del programa, para facilitar su lectura:

```
PROGRAM Circulo(input,output);

CONST
pi              = 3.1415926536;
aviso           = 'Entre el radio:';

VAR
  radio,
  superficie      : real;

BEGIN
  WriteLn;
  write(aviso);
  read(radio);
  superficie:=pi*radio*radio;
  WriteLn;
  WriteLn('La superficie de un circulo',
    'de radio',radio : 8 : 3);
  WriteLn('es:',superficie : 10 : 3)
END.
```

En este ejemplo hay dos aspectos sintácticos que hemos de observar. Primero, la parte VAR declara dos identificadores del mismo tipo, ambos reales. No se necesitan, sin embargo, dos declaraciones separadas, dado que las listas de elementos simplemente se separan mediante una coma; esto es universal en PASCAL. Por consiguiente, cuando especificamos más de un argumento para un procedimiento (como en las sentencias WriteLn) se aplica la misma sintaxis. La segunda característica nueva es el formato de salida que se utiliza para evitar la notación científica por defecto de los valores reales. Opcionalmente, podríamos especificar dos enteros, separados por dos puntos, para forzar una cierta anchura de campo para el número entero y su parte fraccional.

En nuestro programa Circulo, tanto al radio como a la superficie se le concederán tres posiciones decimales. Puesto que la superficie será un número mayor (y, por tanto, más largo), le concedemos un total de 10 posiciones en vez de las ocho que le concedemos al radio. Estos valores enteros deben ser mayores que cero, permitir un posible signo, tener al menos un dígito y acomodar el punto decimal, escrito antes de la parte fraccionaria. Los valores ilegales producirían errores en el momento de la ejecución, o (en el mejor de los casos) harían que el formato saliera en notación científica; WriteLn (X : 6 : 2) no dejaría sitio para números mayores que 99.99, por ejemplo.

Más valioso aún es el hecho de que el PASCAL redondeara automáticamente el último dígito para dar la mayor exactitud en cualquier campo numérico solicitado. Además, se puede utilizar cualquier variable o expresión y no tan sólo una constante. Ello permite una enorme flexibilidad, incluyendo facilidades para tabulación. Con todos los otros tipos de datos, sólo se necesita un valor entero (en realidad, sólo se permite uno) para especificar la anchura del campo. Especificar una anchura de uno haría que los enteros se escribieran en el campo de dimensiones mínimas, sin espacios. Por lo tanto, hemos de recordar colocarlos nosotros mismos si es que han de tabularse los resultados. Por ejemplo:

```
WriteLn('Total:' : 20, peso : 1,'toneladas.')
```

Normalmente el PASCAL se negará a dar una salida confusa o inexacta, pero la capacidad para suprimir todos los espacios implica que debemos tener cuidado de no imprimir dos números consecutivos con un campo de uno. Por ejemplo, si 12 y 34 se escribieran de este modo, la salida sería 1234.

# Hacia el Nuevo Mundo

## Presentamos el proyecto de un juego de simulación basado en un viaje a las Indias durante el s. XVI

Además de proporcionar a los entusiastas de los "marcianitos" juegos de acción rápida que ponen a prueba los reflejos, el ordenador es un medio excelente para un tipo de desafío bastante diferente: el juego de simulación. Éste implica aptitudes de naturaleza muy diferente a las que se requieren para los juegos recreativos. Tal como lo indica la palabra "simulación", al objeto del juego se crea un modelo basado en ordenador del mundo real. Estos modelos pueden ser simuladores en "tiempo real", como los programas simuladores de vuelo o de estrategia.

En el juego de estrategia, el jugador o el grupo de jugadores tiene asignada una serie de tareas que llevar a cabo y se le proporciona información para ayudarlo a tomar las decisiones. Un ejemplo típico es un juego mercantil, en el cual a los participantes se les entregan sumas de dinero y deben comerciar con mercancías con el fin de obtener los máximos beneficios. Lo que hace que esta clase de simulación sea diferente de, pongamos por caso, un juego de aventuras, es el hecho de que mientras el jugador tiende a tropezar con obstáculos y peligros a medida que se va desplazando por el mundo de aventuras y los va sorteando a medida que se le van presentando, en un juego de simulación el jugador a menudo va planificando por adelantado, administrando sus recursos para hacer frente a las contingencias esperadas (y también a las imprevistas).

**Mundos de ficción**
En la fotografía vemos ejemplos de los tres tipos principales de simulación. *Space shuttle* es una realista simulación estilo recreativo de la lanzadera espacial auténtica. *Air traffic control* es una simulación en tiempo real en la cual el jugador ha de impartir instrucciones a varios aviones para que puedan despegar y aterrizar con seguridad. *The great nordic war* es una simulación de la guerra de los Treinta años, asumiendo el jugador el papel del rey de Suecia




Liz Heaney

Este tipo de juego exige a los participantes, por lo tanto, aptitudes de administración y previsión; en otras palabras, capacidad para planificar con antelación en vez de "sobre la marcha". Por supuesto, para reflejar el mundo real en un juego de simulación no se puede permitir que todo salga exactamente tal como se planeó; una buena simulación siempre introducirá factores aleatorios que alteren los esquemas del jugador. Un buen jugador, por consiguiente, se debe ocupar a fondo en la planificación de contingencias, para minimizar el daño que lleguen a infligir los sucesos imprevisibles.

Los juegos de simulación poseen la ventaja de permitir que las personas participen formando un equipo, combinando su talento y sabiduría en contra del ordenador, para alcanzar el objetivo prefijado. Por este motivo los juegos de simulación se están comenzando a utilizar en las escuelas, dado que permiten que el niño modele la clase de comportamiento que cabría esperar de él en el mundo de trabajo de los adultos, combinando los recursos disponibles y los talentos de un pequeño grupo para alcanzar cierto objetivo. Participar en juegos de simulación es, asimismo, sumamente entretenido.

Además de todas estas cualidades, los juegos de simulación cubren un área que muy raramente abordan la mayoría de las aplicaciones para ordenador: se pueden suscitar cuestiones morales, teniendo los jugadores que sopesar, por ejemplo, el aumento de la rentabilidad contra el bienestar de los empleados. Cuando tales decisiones se toman en equipo, suelen ir precedidas de acalorados debates acerca de los pros y los contras.

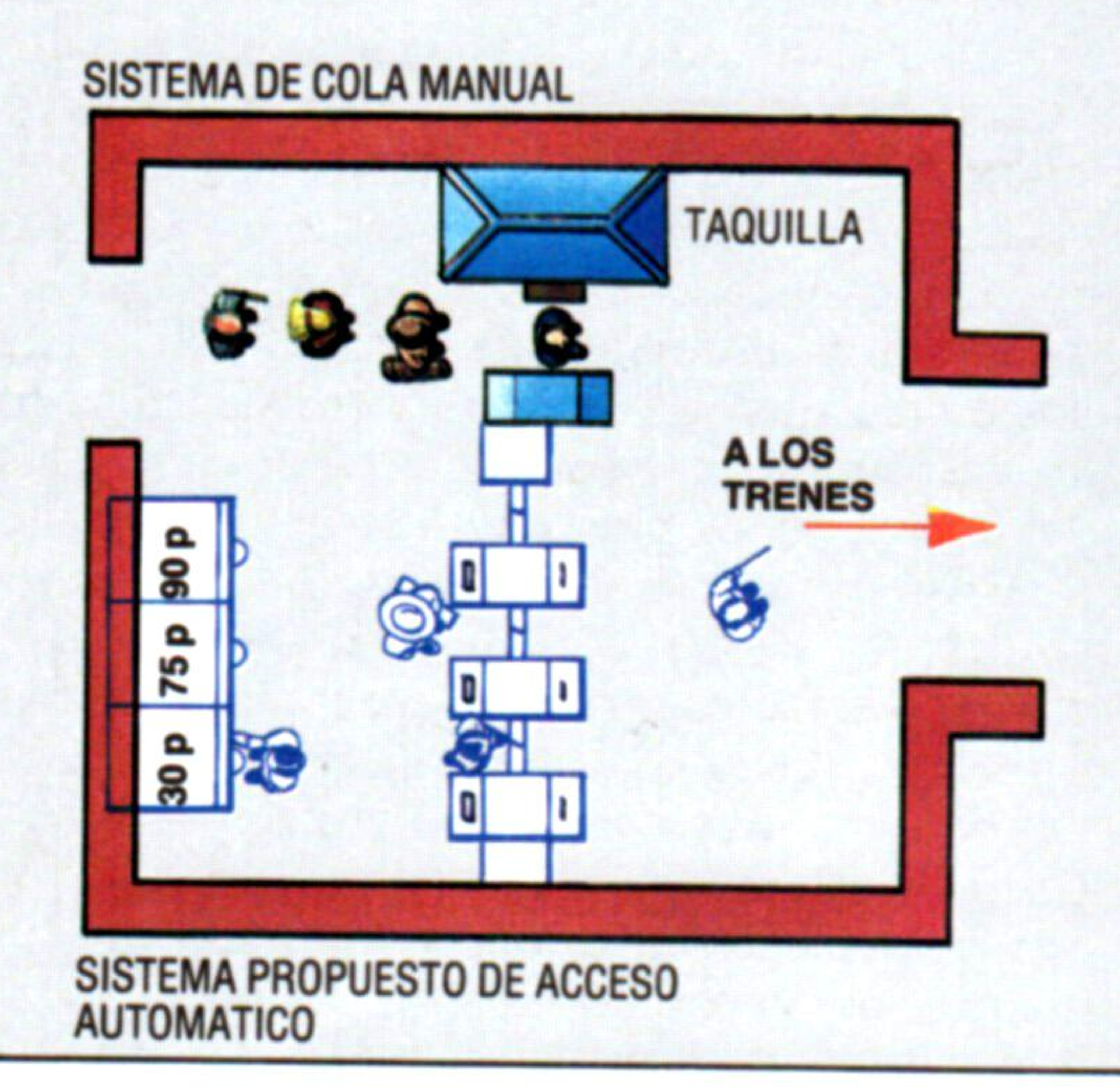
Existen muchos escenarios diferentes que se prestan bien para la simulación por ordenador; a menudo el factor común es que la simulación implica llevar a cabo una tarea específica para alcanzar un objetivo muy concreto. En algunos casos éstos los puede seleccionar el jugador. El objetivo podría ser permanecer en el negocio durante un tiempo dado o, por el contrario, ganar millones de pesetas. Muchas simulaciones se basan en la estrategia financiera, pero algunas poseen objetivos más filantrópicos. Un juego de simulación que se utiliza actualmente en las escuelas primarias de Gran Bretaña es un proyecto para localizar y sacar a la superficie los restos del naufragio del galeón isabelino *Mary Rose*. En este juego a los participantes también se les entregan documentos para complementar el programa y proporcionar pistas adicionales que ayuden al proceso de tomar decisiones.

En esta serie de proyectos de programación construiremos un juego de simulación mercantil, situando a los jugadores en condiciones de tener que equipar y navegar un barco hasta el Nuevo Mundo, en el s. XVI. El juego está construido como una serie de módulos que se encajan entre sí, y el pro-

# Bajo tierra

En la actualidad el uso de las simulaciones está muy generalizado en el campo de la educación y de los juegos, pero las primeras simulaciones basadas en ordenador se desarrollaron como una herramienta de diseño y gestión. Las simulaciones revisten gran valor para el diseño del flujo de tráfico y sistemas de servicios, en los cuales la naturaleza del modelo los hace ideales para la simulación por ordenador. Tales simulaciones por lo general se basan en principios estadísticos y matemáticos que predicen distribuciones de frecuencia de las variables utilizadas.

Podemos demostrar los principios de una simulación analizando un simple sistema de servicios. El sistema existente permite el acceso a la estación de metro mediante la compra de un billete en una taquilla. Se propone instalar una cantidad de barreras de billetes automáticas que operan con billetes adquiridos con anterioridad. El diagrama de flujo ilustra los procesos implicados en la creación de una simulación sencilla del sistema. Se utilizan tres distribuciones: una de llegada de pasajeros, otra de horario de funcionamiento de la taquilla, y una tercera para el método de acceso (es decir, manual o automático). Las dos primeras distribuciones provienen de observaciones estadísticas y métodos matemáticos; la tercera variará a cada ejecución del programa con el fin de observar las ramificaciones de diversas proporciones de puertas de acceso manuales y automáticas. Esta simulación calcula el tiempo de espera promedio de los pasajeros




Comienzo
Inicializar
Ha concluido el tpo.? — S: Calcular y producir results. → Stop; N:
Porción tpo.=tpo. inf. de llegada y servicio de taquillas
Dism. tpo. lleg. en porción tiempo
Dism. tpo. serv. de taquillas en porción tiempo
Incrementar tpo. espera total en porción tiempo × longitud cola
Ha llegado el pasajero? — S: Muestrear distribución entradas → Es automática la entrada? — S: Aumentar el número de accesos al sistema; N: Aumentar cantidad en la cola
Atendido pasajero en taquilla? — S: Incrementar núm. accesos al sist.; N:
Está libre la taquilla? — S: Hay más en la cola? — S: Disminuir cantidad en la cola → Muestrear distribución servicios taquilla; N; N

La simulación no se realiza en tiempo real, sino que calcula el tiempo hasta el siguiente evento cada vez que se ejecuta el bucle. Este "paquete" de tiempo (o "porción de tiempo") se utiliza para actualizar las otras variables de la simulación. El empleo de porciones de tiempo permite que el sistema simulado se ejecute más rápidamente que el sistema real; una hora de la simulación es equivalente a un día del sistema real

grama está diseñado de modo tal que el escenario se pueda modificar con facilidad, proporcionando un esqueleto alrededor del cual se podría construir un contexto diferente. Por lo tanto, la simulación se podría modernizar para convertirse en un viaje de ciencia-ficción, comerciando con un planeta lejano a través de una nave espacial.

El objetivo del juego, para uno o más jugadores, consiste en comerciar con las mercancías obteniendo los máximos beneficios, pero equilibrando esta finalidad con la seguridad y el bienestar de la tripulación. El juego se dividirá en tres etapas: preparación, viaje y comercio. La primera fase supone reunir los recursos y planificar el viaje, y la segunda introduce en el juego elementos al azar. Su habilidad para hacer frente a estas contingencias dependerá en gran medida de lo bien que haya elaborado su plan durante la primera fase del juego. La tercera fase implica establecer buenas relaciones con los nativos y comerciar con ellos de forma rentable.

Desde el punto de vista del programador, la construcción del juego se basará en una serie de módulos independientes y ofreceremos complementos al BASIC para el BBC Micro, el Spectrum y el Commodore 64. Las principales tareas del programa son llevar el registro de las decisiones tomadas por el jugador, crear diversas contingencias y analizar y controlar el estado del jugador a medida que va avanzando el juego. En el próximo capítulo abordaremos el primer módulo del programa.

# Persecución

**Pongamos a prueba su habilidad para atrapar a un amigo de lo ajeno. El programa en BASIC está escrito para el microordenador Alice**

El ladrón ha escapado llevándose el botín. Se esconde en la ciudad, y usted tiene treinta minutos para encontrarlo y detenerlo. Atención, ¡no se precipite! En efecto, si se echa sobre él sin pensar, el fugitivo tiene todas las posibilidades de escapársele. La mejor manera de atraparlo es alcanzándolo de lado. (Es el método más efectivo a condición de no fallar.) Si no se siente lo bastante seguro de sí mismo, atáquelo de frente, lo cual es más fácil pero mucho menos eficaz, ya que no es tan discreto. Otro consejo: no intente perseguirlo; no daría resultado, pues él es mucho más rápido que usted. Ha de observar sus movimientos como si fuera un detective. Cuando vea que da la vuelta, acérquese sigilosamente y sorpréndalo en el momento justo. Pero recuerde: ¡el tiempo va pasando!

Desplazamiento:

Z =arriba
Q =izquierda
S =derecha
W =abajo

```
5 REM ******************************
10 REM *           PERSECUCION          *
15 REM ******************************
19 REM S=PUNTOS
20 S=0
25 REM V$=CARACTER LADRON
30 V$=CHR$(128)
35 REM P$=CARACTER JUGADOR
40 P$=CHR$(191)
50 GOSUB 2000
104 REM
105 REM BUCLE PRINCIPAL
106 REM
109 REM MOVIMIENTO JUGADOR
110 D$=INKEY$
120 D=(D$="Q")-(D$="S")+32*((D$="Z")-(D$="W"))
130 IF D<>0 THEN D0=D
135 REM DESCONTAR TIEMPO
150 T=T-0.1
160 PRINT@ 480,"TIEMPO  :";INT(T+1);
165 REM TIEMPO CONSUMIDO?
170 IF T<0 THEN 490
180 P=P+D0
190 C=PEEK(16384+P)
195 REM LADRON ATRAPADO?
200 IF C=128 THEN 4000
205 REM OBSTACULO?
210 IF C<>159 THEN P=P1
220 PRINT@ P1,CHR$(159);
230 PRINT@ P,P$;
240 P1=P
245 REM DESPLAZAMIENTO LADRON
250 V=V+DV
255 REM OBSTACULO?
260 IF PEEK(16384+V)<>159 THEN GOSUB 600
270 IF PEEK(16384+V)<>159 THEN 250
280 PRINT@ V1,CHR$(159);
290 PRINT@, V,V$;
300 V1=V
310 GOTO 110
484 REM
485 REM FIN
486 REM
490 D$=INKEY$
500 IF R<S THEN R=S
510 PRINT@ 166,"TIEMPO CONSUMIDO";
520 PRINT@ 234,"PUNTOS  :";S;
530 PRINT@ 266,"RECORD  :";R;
540 PRINT@ 326,"OTRA  ?";
550 D$=INKEY$
560 IF D$="" THEN 550
570 IF D$<>"N" THEN 20
580 END
594 REM
595 REM OBSTACULO
596 REM
600 D2=D2+1
610 GOSUB 900
620 IF PEEK(16384+V1+DV)=159 THEN
    V=V1+DV:RETURN
630 D2=D2-2
640 GOSUB 900
650 IF PEEK(16384+V1+DV)=159 THEN
    V=V1+DV:RETURN
660 D2=D2-1
670 GOSUB 900
680 V=V1+DV
690 RETURN
900 IF D2>4 THEN D2=D2-4
910 IF D2<1 THEN D2=D2+4
920 DV=(D2=1)-(D2=3)+32*((D2=2)-(D2=4))
930 RETURN
1994 REM
1995 REM INICIALIZACION
1996 REM
2000 CLS 2
2005 REM TRAZAR CUADRADO
2020 FOR I=0 TO 31
2030 PRIT@ I,CHR$(175);
2040 PRINT@ 448+I,CHR$(175);
2050 NEXT I
2060 FOR I=1 TO 13
2070 PRINT@ I*32,CHR$(175);
2080 PRINT@ I*32+31,CHR$(175);
2090 NEXT I
2095 REM INSTALACION OBSTACULOS
2100 FOR I=1 TO 70
2110 GOSUB 3000
2130 PRINT@ P,"  ";
2140 NEXT I
2145 REM CUADRO LADRON
2150 GOSUB 3000
2160 V=P
2170 PRINT@ V,V$;
2180 V1=V
2185 REM CUADRO JUGADOR
2200 GOSUB 3000
2210 PRINT@ P,P$;
2220 P1=P
2230 T=30
2240 D0=0
2250 DV=0
2260 D2=0
2270 RETURN
2994 REM
2995 REM POSICION ALEATORIA
2996 REM
3000 P=RND(414)+32
3005 REM POSICION OCUPADA?
3010 IF PEEK(16384+P)<>159 THEN 3000
3020 RETURN
3994 REM
3995 REM GANA
3996 REM
4000 FOR I=1 TO 5
4005 REM SIRENA
4010 SOUND 35,10
4020 SOUND 5,10
4030 NEXT I
4040 S=S+1
4050 GOTO 50
```



# Ritmo y melodía

## El CX5M aprovecha la experiencia de Yamaha en sonido y se convierte en el primer ordenador personal exclusivo para música

Aunque adherido al estándar MSX, que permite su utilización para aplicaciones tales como juegos o proceso de textos, el ordenador Yamaha CX5M está decididamente dirigido al amante de la música electrónica. Se puede conectar un teclado de piano externo YK-10 o YK-01 en un conector situado a uno de los lados de la máquina, donde hay asimismo un par de puertas MIDI (interface digital para instrumentos musicales) que posibilitarán la ejecución desde el ordenador de cualquier sintetizador o dispositivo para música electrónica con una interface MIDI. Hay igualmente un par de conectores microjack para el empleo de altavoces externos.

El ordenador tiene un aspecto muy parecido al de las otras máquinas de la gama MSX. Hay 48 teclas de máquina de escribir rodeadas por 10 teclas de función, que contienen características estándares MSX, tales como un retroceso con borrado, una tecla Graph para visualizar un conjunto de caracteres para gráficos, y una tecla Code, la cual, al ser pulsada junto con una de las teclas de máquina de escribir, imprime toda una variedad de caracteres de idiomas extranjeros. Encima del teclado hay cinco teclas de función, capaces de proporcionar 10 funciones programables. En el encendido, éstas pasan por defecto a las funciones MSX estándares de AUTO, LIST, RUN, etc. En el extremo inferior derecho del teclado está el grupo de las cuatro teclas del cursor y encima del mismo hay un conjunto de cinco teclas que configuran otras instrucciones estándares MSX tales como INSertar, DEL (eliminar) y STOP. Encima del teclado hay una puerta para cartucho.

## La arquitectura de la máquina

En el lado derecho del CX5M hay un par de puertas para palanca de mando tipo Atari. En su parte posterior, el ordenador posee un conector marginal en paralelo, la interface para impresora compatible con Centronics, una puerta para cassette, un enchufe hembra de sonido para salida mono, un enchufe hembra para monitor de video compuesto, un enchufe para televisión RF y la entrada de la fuente de alimentación eléctrica. Una única caja contiene las puertas MIDI y la interface del teclado, que les otorgan al CX5M sus capacidades musicales exclusivas. Esta caja se introduce en un conector marginal situado debajo de la máquina, al lado izquierdo. La razón de ello reside en que Yamaha pretende producir una serie de interfaces con características diferentes. Cuando estos dispositivos salgan a la venta, los usuarios podrán intercambiar las interfaces simplemente quitando un tornillo y colocando en el conector marginal la interface nueva.

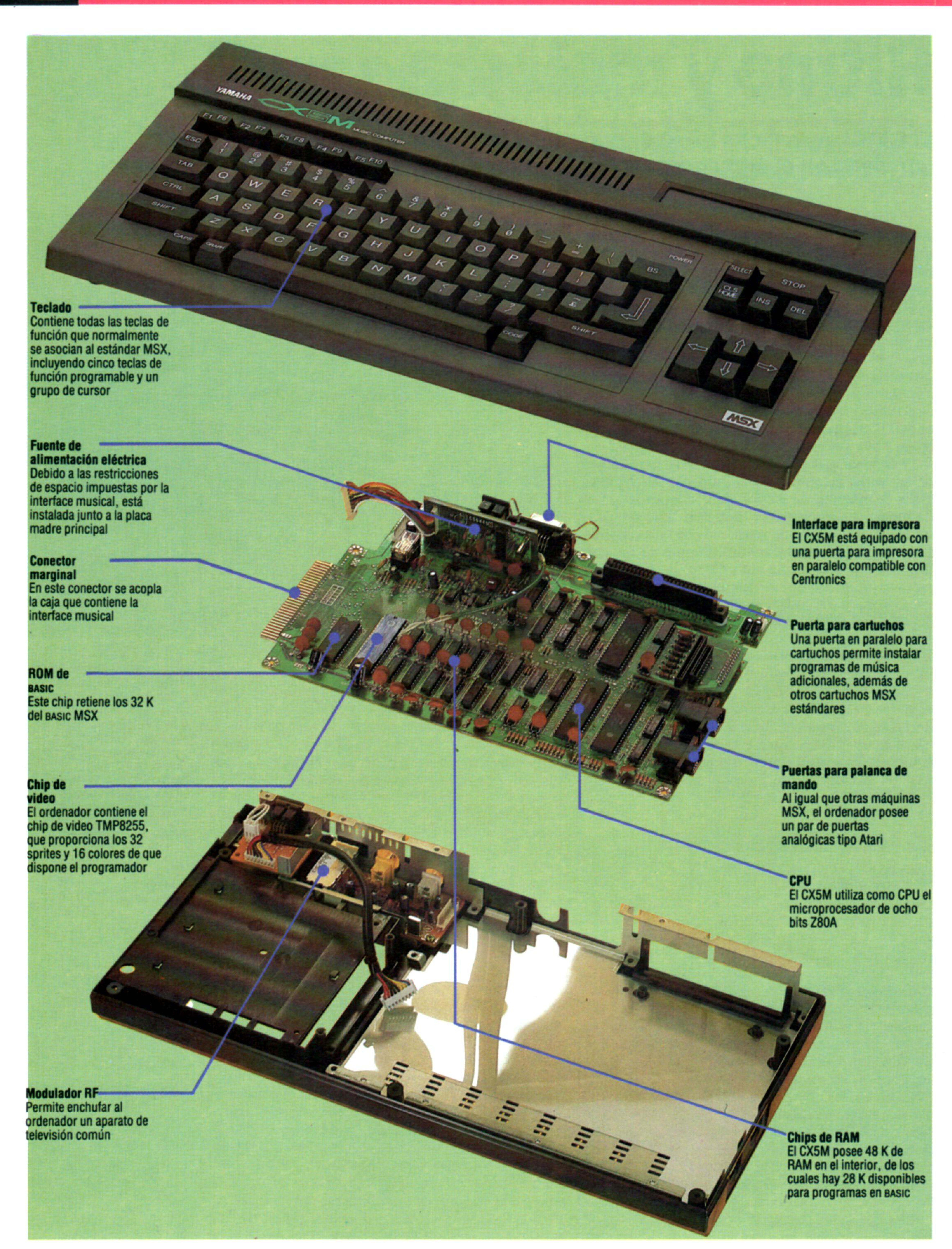


Chris Stevens

**Cajas de música**
Se prevé que el Yamaha CX5M se venderá mejor en las tiendas de instrumentos musicales que en las de informática. Para realzar sus capacidades musicales, el ordenador se comercializa como un paquete junto con un teclado de piano ya sea YK-01 o bien YK-10. El teclado de piano se enchufa en una pequeña caja para interface de música que está atornillada en la parte inferior del ordenador

El teclado YK-01 de tres octavas y media (que se vende con el ordenador y que podemos apreciar en la fotografía) posee teclas con un agradable tacto profesional, si bien los músicos de teclado serios las encontrarán demasiado pequeñas como para resultar totalmente cómodas. Utilizando el software propio del ordenador, el teclado se puede "dividir", lo que significa que una voz programada se puede tocar en la parte inferior del teclado mientras en el resto de las teclas se puede tocar otra completamente diferente. Esto significa, por ejemplo, que uno puede tocar "cuerdas" en las teclas superiores mientras proporciona con las inferiores un acompañamiento de "contrabajo". El teclado también puede funcionar en modalidad monofónica o polifónica (o ambas), tocando simultáneamente hasta ocho notas.

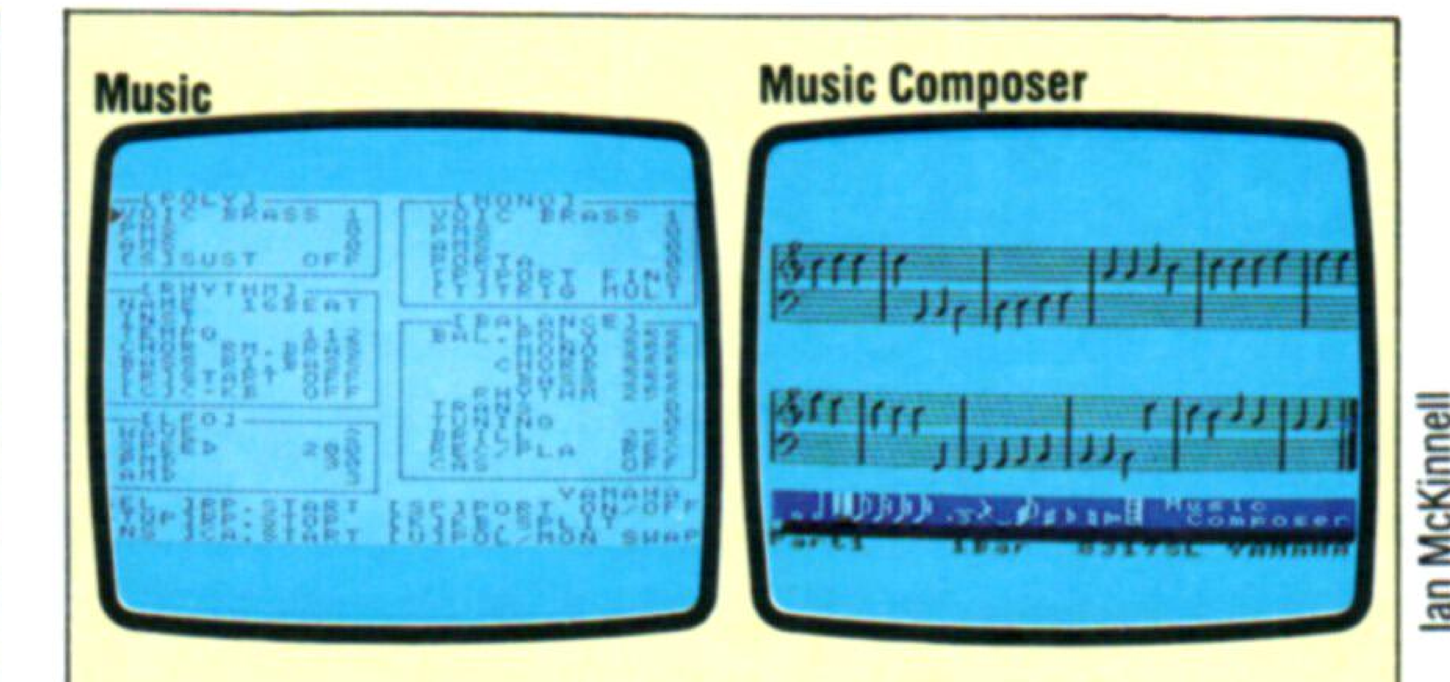
Tras el encendido, se visualiza la pantalla azul estándar MSX. Se puede acceder al programa de música mediante la instrucción CALL MUSIC, la cual visualizará un menú con cinco bloques de las diversas opciones disponibles. El usuario puede entonces pasar la lista hacia abajo mediante el empleo de la tecla Return, y los parámetros de cada opción se alteran pulsando las teclas del cursor. Los bloques etiquetados POLY y MONO permiten que el músico seleccione cuál de las 46 voces preprogramadas se debe utilizar, y en cuál de las dos modalidades. Estas voces van desde instrumentos musicales convencionales (como órgano, guitarra e instrumentos de percusión como el vibráfono y el cencerro) hasta una gama de sonidos cotidianos (como una "sirena de ambulancia" y "sonido de gotas de lluvia"). Debido a la naturaleza de algunas voces, hay algunas


YAMAHA
CX5M
MSX
**Teclado**
Contiene todas las teclas de función que normalmente se asocian al estándar MSX, incluyendo cinco teclas de función programable y un grupo de cursor
**Fuente de alimentación eléctrica**
Debido a las restricciones de espacio impuestas por la interface musical, está instalada junto a la placa madre principal
**Conector marginal**
En este conector se acopla la caja que contiene la interface musical
**ROM de BASIC**
Este chip retiene los 32 K del BASIC MSX
**Chip de video**
El ordenador contiene el chip de video TMP8255, que proporciona los 32 sprites y 16 colores de que dispone el programador
**Modulador RF**
Permite enchufar al ordenador un aparato de televisión común
**Interface para impresora**
El CX5M está equipado con una puerta para impresora en paralelo compatible con Centronics
**Puerta para cartuchos**
Una puerta en paralelo para cartuchos permite instalar programas de música adicionales, además de otros cartuchos MSX estándares
**Puertas para palanca de mando**
Al igual que otras máquinas MSX, el ordenador posee un par de puertas analógicas tipo Atari
**CPU**
El CX5M utiliza como CPU el microprocesador de ocho bits Z80A
**Chips de RAM**
El CX5M posee 48 K de RAM en el interior, de los cuales hay 28 K disponibles para programas en BASIC






Éstos son algunos de los cartuchos que han salido a la venta para el CX5M. Cada uno de los programas que vemos le permite al ordenador llevar a cabo una función diferente, desde programar al sintetizador DX-7 hasta componer programas de música. Cada cartucho viene con su propio manual. Sin embargo, son relativamente caros y los usuarios quizá prefieran esperar hasta que el precio se reduzca o aparezca software perfeccionado




Arriba vemos dos aplicaciones para el CX5M. El paquete Music está incorporado en el ordenador y se accede a él a través de la instrucción CALL MUSIC. Desplazando el cursor con las teclas Function y Return, los parámetros indicados se pueden alterar luego pulsando las teclas del cursor. El Music Composer permite al usuario escribir una partitura musical en la pantalla, que se puede editar y enviar después a la impresora para obtener una salida impresa

Ian McKinnell

que no están disponibles cuando se utiliza el "sostenido". La alteración de las voces de las modalidades POLY y MONO le permite al usuario establecer tonos contrastantes para usar con el teclado dividido; es decir, designar algunas teclas como monofónicas y otras como polifónicas. Es posible, por supuesto, establecer todo el teclado en cualquiera de las modalidades; sin embargo, no se lo puede dividir en dos secciones POLY.

Los sonidos que se pueden producir son notables, aun cuando se emitan a través de un altavoz de televisor normal, si bien algunos de los sonidos guardan poco parecido con los nombres que se les ha dado. Por ejemplo, "tren" suena simplemente como un órgano eléctrico normal. Éste es un defecto común de los fabricantes de sintetizadores; pero, siempre y cuando el usuario no espere una reproducción exacta de los sonidos cotidianos, la mayoría de los músicos se sentirán más que satisfechos con las voces proporcionadas.

Otro bloque, RHYTHM, proporciona un acompañamiento rítmico a una melodía. Son seis los diferentes ritmos disponibles, que se pueden construir utilizando cuerdas, contrabajo y percusión. De éstos, los tonos de contrabajo y cuerdas los puede establecer el músico. El tiempo del ritmo se puede variar y hay asimismo una opción que le permite al músico variar la altura *(pitch)* del ritmo desde el teclado de piano. Alterando la forma de la onda de sonido mediante el bloque LFO (Low Frequency Oscillator), se pueden modular tonos para producir sonidos diferentes. El bloque BALANCE permite el ajuste de volumen independiente de cada elemento rítmico, más las modalides POLY y MONO. Por lo tanto, se puede reducir el acompañamiento rítmico de fondo y elevar el volumen de la modalidad POLY para darle mayor énfasis a la melodía. Cuando el músico se sienta satisfecho con los sonidos producidos, los datos se pueden guardar en cassette y cargar posteriormente. Esto le ahorra al músico el tener que escribir los distintos parámetros que han sido seleccionados y volver a establecerlos después.

## Programas para el Yamaha CX5M

Aparte del software incorporado, Yamaha ha lanzado una serie de cartuchos para utilizar con el CX5M. Esta gama de cartuchos incluye el FM Voicing Program, que es una versión ampliada del software incorporado. El programa le permite al músico manipular de una forma más cabal la forma de la voz mediante el ajuste de las frecuencias y los algoritmos con los que fue construida. Un segundo programa de voz en cartucho le permite al ordenador programar el sintetizador Yamaha DX7. El FM Music Macro permite programar el ordenador con un conjunto de instrucciones adicionales en BASIC, mientras que el FM Music Composer visualiza un pentagrama en blanco y permite entrar una partitura convencional y volcarla luego a una impresora externa.

Aunque podría parecer que Yamaha ha proporcionado un amplio juego de programas para el CX5M, uno se queda con la sensación de que el software no explota totalmente las capacidades de la máquina. Los programas de modulación, en especial, parecen un poco limitados; en ocasiones resulta difícil discernir cualquier cambio perceptible en el sonido de una nota. Esto es un hecho inusual, puesto que Yamaha ha basado su sistema operativo en el utilizado con el sintetizador DX7, cuya enorme gama de sonidos hace que, en comparación, el CX5M parezca insignificante. Además, aunque los métodos elegidos por Yamaha para alterar los parámetros de los bloques sean adecuados, en ocasiones parecen engorrosos; por ejemplo, utilizar las teclas del cursor para alterar parámetros mientras se mueve el cursor con la tecla Return. No obstante, Yamaha afirma que el software actualizado ya está en camino.

El CX5M está obviamente dirigido a quienes desean un ordenador personal y posean, al mismo tiempo, un interés por la música electrónica. Para estas personas, el paquete que comentamos es, indudablemente, tentador. Uno no sólo obtiene un sistema MIDI completo para ordenador y un teclado, sino que el CX5M ofrece además un potencial mucho mayor que un sistema de precio similar basado, por ejemplo, en el Commodore 64. Parece, no obstante, que el CX5M está destinado a convertirse en un ordenador "culto". El desembolso económico que supone su adquisición es difícil justificarlo ante alguien que no se interese por la música electrónica, puesto que ahora se puede adquirir un ordenador MSX similar a menos de la mitad de precio.

El éxito que obtenga la firma Yamaha con el establecimiento del ordenador como la máquina que deben adquirir los amantes de la música, depende del futuro desarrollo de su software y sus interfaces.

## YAMAHA CX5M

**DIMENSIONES**

413×216×64 mm

**CPU**

Z80A operando a 3,58 MHz

**MEMORIA**

48 K de RAM, de los cuales hay 28 K disponibles para programas en BASIC

**PANTALLA**

Pantalla para textos de 40×24, pantalla para gráficos de 256×192 pixels, con 16 colores y hasta 32 sprites

**INTERFACES**

Impresora Centronics, TV, monitor compuesto, salida de audio, salida estéreo, 2 puertas para palanca de mando, puerta para cassette, puerta para cartuchos de ROM, bus de ampliación, interface para teclado, puertas para entrada y salida de MIDI

**LENGUAJES DISPONIBLES**

BASIC, PASCAL, Assembly

**TECLADO**

67 teclas tipo máquina de escribir con grupo de cursor, más 5 teclas de función programables. El teclado de piano es polifónico de ocho notas con una gama de 3,5 octavas

**DOCUMENTACION**

El manual es muy irregular. Si bien ofrece algunos detalles sobre cómo utilizar el software para música, no ofrece ninguna explicación de guía. El BASIC MXS apenas si se menciona

**VENTAJAS**

La máquina ofrece una valor excelente para el músico electrónico. Considerado como un instrumento musical, el CX5M no tiene competencia dentro de su gama de precio

**DESVENTAJAS**

Será necesario mejorar mucho el software para que la máquina llegue a alcanzar un éxito a largo plazo. El manual es muy limitado y la máquina es demasiado cara para quienes no estén interesados en la música electrónica

# Lectura de la Biblia

## "'The Word' processor" (El procesador de "La Palabra") es un programa que resalta esclarecedoramente el valor de las grandes bases de datos

**"The Word" Processor (El procesador de La Palabra)**
para máquinas MS-DOS
**Distribuido por**
Access Software Ltd, 36 Aybrook Street, London W1M 3JL, Gran Bretaña
**Autores**
Bible Research Systems, Austin, Texas (Estados Unidos)
**Formato**
Discos

Ciertas ocupaciones y estudios escolásticos exigen una búsqueda constante a través de grandes cuerpos de texto. Los estudiantes de la Biblia, por ejemplo, requieren de concordancias (índices especializados) para ayudar al estudio de los 39 libros del Antiguo Testamento y los 27 libros del Nuevo Testamento: aproximadamente 750 000 palabras en total.

Los ordenadores por lo general son especialmente indicados cuando es necesario efectuar búsquedas en cantidades tan ingentes de texto, pero a menudo las limitaciones de memoria y capacidades de los discos restringen esta aplicación en los micros personales. Es evidente, entonces, que un programa escrito especialmente podría simplificar muchísimo la tarea de realizar búsquedas de los textos bíblicos apropiados. El programa que nos ocupa, al que se le ha dado el desenfadado nombre de *"The Word" processor* (El procesador de La Palabra), no sólo permite buscar referencias específicas, sino que también construye índices que reducen el tiempo que puede ocupar una búsqueda.

La magnitud de la tarea asumida por los programadores de *"The Word" processor* se hace evidente cuando se abre el paquete. El manual es un delgado folleto de 36 páginas, pero el programa completo, incluyendo archivos de textos bíblicos, se compone de siete discos de doble cara, todos los cuales han de ser copiados. Si se emplea una unidad de una sola cara y 40 pistas, entonces habrá que manipular un total de 14 discos separados, lo que demuestra a todas luces que la manipulación de tantos discos de 5 ¼ pulgadas es demasiado complicada para resultar verdaderamente eficaz.

*"The Word" processor* requiere una instalación cuidadosa. Es razonable suponer que la mayoría de los estudiantes no son necesariamente apasionados de la informática, a pesar de lo cual el programa en BASIC es bastante engañoso en la forma en que se ejecuta: cuando lo utilizamos, dejó de responder al MS-DOS y respondió sólo al PC-DOS. Las pistas del intérprete de BASIC y del sistema de la máquina del disco DOS se deben copiar en el disco del programa. El estudiante debe entonces elegir entre ejecutar (RUN) el programa principal (TWP), si está utilizando un IBM-PC, otra versión para máquinas compatibles con IBM (ATWP), o una versión especial para ordenadores con sólo 64 K de memoria (TWP64). El usuario ya experimentado supondrá con razón que, una vez instalados en el disco las pistas del sistema, el BASIC y los programas pertinentes, se producirá la autocarga; pero no hay ningún programa AUTOEXEC. BAT con este objeto.

Una vez cargado y en ejecución, la página de títulos nos hace comprender que éste no es un paquete de software común, dado que, además de la nota habitual de protección del software ("Este software está protegido por las leyes de *copyright* de Estados Unidos. Su reproducción o la utilización de copias no autorizadas está penada con prisión o multas de hasta 10 000 dólares"), hay asimismo una cita bíblica muy apropiada:

Exodo 20:15 "No robarás."

Este precepto va seguido por el menú de apertura:

```
<F1> —   VISUALIZAR INSTRUCCIONES DE AYUDA
<F2> —   ESTABLECER OPCIONES DE CONTROL
<F3> —   LA MODALIDAD DE IMPRESION ESTA
         APAGADA. ENCENDERLA
<F4> —   GAMA:
<F5> —   FIN DE ESTA SESION
<F6> —   CREAR UN INDICE
<F7> —   VISUALIZAR O MODIFICAR INDICE
<F8> —   MEZCLAR INDICES
<F9> —   ELIMINAR UN INDICE
<F10> —  VISUALIZAR TEXTO DE LAS ESCRITURAS
```

<F2> produce un submenú de nueve opciones de control, incluyendo anchura de pantalla (40 u 80 caracteres), tabulaciones para los márgenes izquierdo y derecho de la impresora, espaciado de líneas, número de líneas por página, dimensiones máximas del índice, visualización de un texto cada vez o una pantalla de textos entera con indicación de los versículos del contexto, y los nombres de todos los índices existentes en el disco actual.

La opción <F4> es una forma de administrar tanto la inmensidad de la Biblia como la proliferación de discos en un sistema basado en discos flexibles. Utilizando una designación de tres letras para cada libro de la Biblia (de *Gen* a *Rev*), y la referencia de capítulo y versículo, uno puede definir el comienzo y el final del parámetro de búsqueda. Si el usuario desea leer un libro específico se pueden omitir las referencias de capítulo y versículo y el programa supondrá que la gama requerida es el comienzo del primer capítulo y el final del último.

Después de haber insertado el disco adecuado, se puede hojear un capítulo (visualizando el primer versículo de la gama mediante <F10>) e ir hacia atrás o hacia adelante mediante las teclas del cursor, que aparecen a través de la parte inferior de la pantalla. La opción de control <F8> cambia la visualización de un versículo cada vez por una pantalla entera de textos, permitiendo leer en su contexto un versículo específico.

El criterio de búsqueda se puede establecer mediante <F2>, la opción SCAN. Tal como sucede con las búsquedas en la mayoría de las bases de datos, se pueden buscar todas las ocurrencias, incluso dentro de otras palabras (de manera que *man* también cogería *demand*), o se puede limitar la búsque-

da sólo a un vocablo específico. También se pueden utilizar máscaras, que permiten buscar todas las palabras que empiecen o terminen con una serie de búsqueda especificada. Una vez establecidas, las referencias de todos los criterios se visualizarán de una en una, ya sea en el contexto o individualmente, según lo que se haya solicitado.

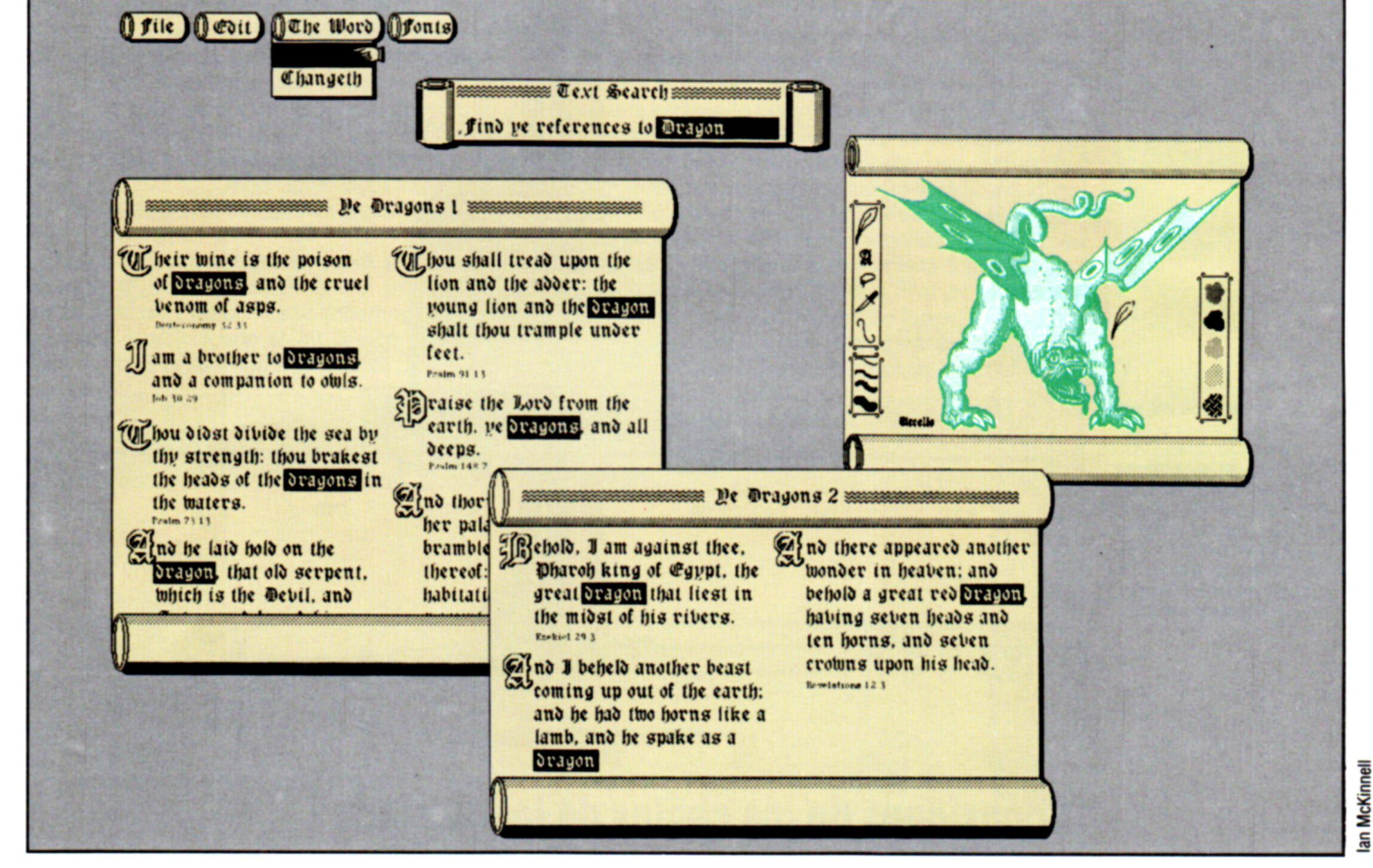
Con la opción <F6> se pueden crear índices, ya sea automáticamente o de forma manual. Los índices también se pueden editar (p. ej., suprimiendo referencias no deseadas), y se pueden mezclar dos o más entre sí, siempre que no exceda la cantidad límite de 1 020 elementos en el índice. Este límite se puede aumentar a 53 040 con la opción <F2>, ESTABLECER OPCIONES DE CONTROL, del menú principal. Si un índice posee más de 1 020 referencias, se lo almacena (utilizando el submenú OPCIONES DE CONTROL) por secciones, y entre las entradas 1 020 y 1 021 habrá una pausa cuando la sección siguiente del índice se lea (READ) del disco a la memoria. También se puede crear un índice de los temas que contiene cada versículo. Esto obviamente se ejecuta bajo el control del usuario, quien define el título del tema y el versículo en el cual se pueda hallar.

Además de la dificultad de manipular cantidades tan grandes de texto, el problema principal de *"The Word" processor* reside en que, dado que se basa en la versión del rey Jacobo, que data de 1611, muchas de las palabras utilizadas en esta magnífica traducción han sufrido un cambio radical de significado en los tres siglos y medio posteriores. De acuerdo a los traductores de la Revised Standard Version de 1952, hay más de 300 de tales palabras y frases; y eso sin incluir los errores de traducción que quedaron patentes merced a la aparición de textos más fiables desde el s. XVII.

Afortunadamente, para evitar este inconveniente se puede utilizar un programa, *The Greek transliterator* (El transliterador griego), que será de ayuda al menos en lo que se refiere al Nuevo Testamento. Este programa permite buscar a través del texto griego las apariciones de una palabra dada de la versión del rey Jacobo, solicitando la visualización de la palabra griega original (no en ortografía griega, sino en caracteres romanos) con una definición de la palabra griega y, si fuera necesario, la frecuencia con la que se utilizó tal palabra y los distintos significados que asumió en diversos contextos. Los índices creados con el TWP también se pueden utilizar con *The Greek transliterator*, y también se pueden crear índices de palabras griegas. El precio de *The Greek transliterator* es el mismo que el del TWP.

Hasta la fecha no se ha producido ningún programa transliterador para el Antiguo Testamento, si bien en Jerusalén se ha programado un gran ordenador para almacenar el Talmud y otras obras religiosas. Posiblemente ésta sea la forma más práctica de emplear los estudios informatizados de la Biblia: como una base de datos de acceso público y no como un sistema individual de recuperación de textos. No obstante, las técnicas empleadas en *"The Word" processor* son interesantísimas y, presumiblemente, se podrían aplicar a otras obras a gran escala: la producción escénica y poética de Shakespeare, por ejemplo, o incluso las obras completas de Marx y Engels.

**El rey Jacobo y el dragón**
Al solicitar *dragon* a través del teclado, la base de datos *"The Word" processor* localizó, entre otras, estas referencias. Actuando como lo haría una concordancia escrita, *"The Word" processor* simplifica la búsqueda de ideas bíblicas relacionadas, símbolos y personalidades, en la versión de la Biblia del rey Jacobo




Ian McKinnell

# En un cable

## Con el fin de controlar el robot, vamos a construir el cable que lo conecta a la interface diseñada en el capítulo anterior

La última tarea en la construcción de la interface es soldar un cable plano de 12 vías desde la placa de la interface hasta un conector tipo D que se enchufará en el robot. Las ocho líneas de datos desde la puerta de E/S están disponibles desde el bus de datos a través del sistema de circuitos de la interface. A la alimentación de potencia de 12 V y 5 V también se accede directamente desde la puerta para ampliación del Spectrum. Ello significa que no tenemos que proporcionar otra fuente de alimentación eléctrica para activar el robot. El suministro de potencia del ZX es capaz de abastecer al ordenador y al robot al mismo tiempo.

Confeccione el cable plano pelando cuatro cables del cable plano especificado en la lista de componentes que ofrecimos para el proyecto. Pele y estañe ambos extremos de cada cabo hacia la placa y el conector tipo D tal como se indica en el diagrama. Compruebe a conciencia su trabajo, asegurándose particularmente de que las alimentaciones de 12 V y 5 V lleguen a las patillas correctas del conector tipo D.

**Tendiendo un cable plano de 12 vías**
Realice el cableado del conector tipo D con la placa de la interface tal como muestra la ilustración. El cableado es directo: los cables se alternan entre las filas de patillas superior e inferior del conector tipo D

E D7 D6 D5 D4 D3 D2 D1 D0 E

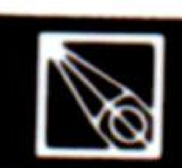
LADO DE COMPONENTES DE LA PLACA DE LA INTERFACE

CABLE PLANO DE 12 VÍAS

12 V D6 D4 D2 D0 E
5 V D7 D5 D3 D1 E

CONECTOR TIPO D

**Conexiones de los cables de la interface**



La interface está ahora completa y podemos enchufarla en el Spectrum para probarla. Con la potencia apagada, enchufe cuidadosamente la placa de la interface en la puerta para ampliación de la parte posterior del Spectrum. La placa se debe insertar de modo que el lado de los componentes quede arriba; si ha insertado el enchufe anulador en la posición 5 del conector de la puerta para ampliación, será imposible colocar la placa de cualquier otra forma. El encaje de la placa puede ser apretado mediante un ligero movimiento de vaivén, a izquierda y derecha, la ranura del PCB de la puerta de ampliación encajará ceñidamente en el conector de la placa de la interface. Enchufe el cable plano en el robot y encienda la potencia. Si todo está bien, en la pantalla aparecerá el mensaje del *copyright* de Sinclair. Ahora estamos en condiciones de probar un programa de verificación.

## Control de motores

El robot se asocia en la puerta de E/S 31, utilizándose sus ocho bits para controlar los motores y recibir entradas provenientes de los sensores del robot. Los cuatro bits inferiores controlan el movimiento. Los bits 1 y 2 controlan la dirección de rotación de los dos motores paso a paso utilizados con el robot. Las cuatro direcciones se pueden seleccionar estableciendo estos dos bits en diversas combinaciones. El bit 3 es el bit de impulsos. Cambiándolo de 0 a 1 ambos motores efectúan un giro de un paso en la dirección que se haya especificado en el correspondiente bit de dirección. El bit 0 es el bit de puesta a cero y normalmente está a 1.

Podemos comprobar la salida de la interface digitando y ejecutando el programa "Controlador del robot", que permite controlar el robot desde el teclado. Las letras T, B, F y H corresponden a adelante, atrás, izquierda y derecha, respectivamente.

La rutina de inicialización establece cuatro variables, que corresponden a las cuatro direcciones posibles del robot. Sus valores se determinan mediante las cuatro combinaciones de los bits 1 y 2 de la puerta de E/S 31. Observando estas variables y sus valores binarios de cuatro bits (véase tabla), podemos ver por qué.

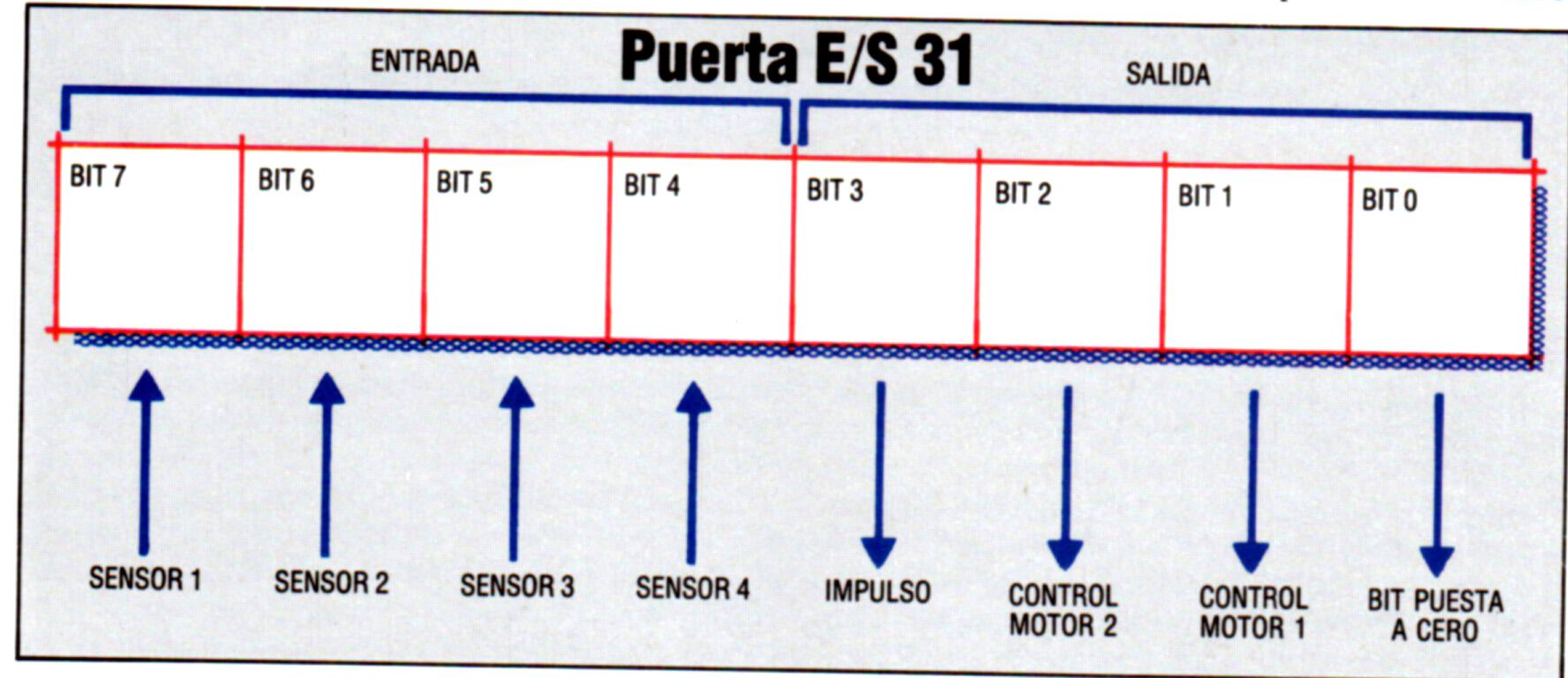
La dirección que toma el robot se retiene en la variable dr. Para conseguir que el robot se mueva en la dirección especificada debemos impulsar los motores estableciendo el bit 3 *high* y luego *low*. Esto lo hace la subrutina de la línea 2000, utilizando OUT, de forma similar a POKE, para colocar un valor en la puerta de E/S 31. El valor en sí mismo se calcula como la dirección actual, dr, más 8 para establecer el bit 3 en 1, es decir, dr+9. Para apagar el bit 3 simplemente no sumamos en el 8, lo que nos da dr+1. Al hacer que el bit 3 se ponga *high* y después otra vez *low*, los motores efectúan un giro de un paso de 7,5°. Como esto corresponde a un movimiento de menos de 1 mm en la rueda, el proceso de impulso se coloca dentro de un bucle para repetirlo un cierto número de veces, determinado por la variable m.

| Dirección | Variable | Valor decimal | Valor binario |
|---|---|---|---|
| Adelante | fw | 4 | 0100 |
| Atrás | bw | 2 | 0010 |
| Izquierda | lf | 6 | 0110 |
| Derecha | rt | 0 | 0000 |

La dirección se altera efectuando una pulsación en el teclado, modificando el valor de dr. Cuando los motores se impulsan la vez siguiente, los bits de dirección del motor cambian haciendo que el motor se mueva en otra dirección.

## Detección de la entrada

Los cuatro bits superiores se utilizan para detectar entradas de los sensores del robot, correspondiendo cada bit a un conector del sistema de parches de la tapa del robot. Normalmente estos bits se mantienen *high* (es decir, poseen un valor de 1), a menos que sean dirigidos a tierra mediante el cierre de un interruptor, en cuyo caso el valor del bit pasa a 0. Para detectar las entradas independientes de cada bit necesitamos un método de aislar cada uno de los cuatro bits superiores y comprobar sus valores. En la mayoría de las versiones de BASIC ello es posible utilizando la instrucción lógica AND para enmascarar cualquier bit en el que no estamos interesados. El BASIC Spectrum posee una instrucción AND, pero no es suficientemente sofisticada como para enmascarar los bits individuales de un número. Sin embargo, el juego de instrucciones Z80 contiene una operación AND que realizará la tarea que necesitamos. Por consiguiente, hemos de escribir un trozo sencillo de código de lenguaje máquina que lleve a cabo el AND lógico en dos números y devuelva el resultado. Para emplear la rutina debe-




**Los bits de la acción**
La interface para el Spectrum establece los cuatro bits inferiores de la puerta de E/S 31 en salida y los cuatro bits superiores en entrada. Los bits de salida controlan las funciones de los motores paso a paso, utilizándose los cuatro bits superiores para aceptar las entradas sensoriales

mos colocar (POKE) los números que deseamos relacionar con AND en dos posiciones de la memoria. El resultado se devolverá mediante la instrucción USR. Este método indirecto nos permite leer cada línea de entrada de forma independiente.

El programa "Parachoques Spectrum" comprueba la acción de la entrada de la interface enviando el robot hacia adelante hasta que encuentre un objeto, retrocediendo entonces a su posición de partida inicial.

Para poner al día a los usuarios del Spectrum, incluimos también los correspondientes *Complementos al BASIC* para algunos de los otros programas que hemos ido creando en la serie de aplicaciones de robot. Los lectores habrán de realizar las alteraciones necesarias sobre las versiones para el Commodore 64. Quizá los usuarios del Spectrum deseen, asimismo, sustituir los nombres de variables en minúsculas por los nombres en mayúsculas utilizados en las versiones Commodore.

## Controlador del robot

```
10 REM **** controlador del robot Spectrum ****
20 GO SUB 1000:REM inicializar
30 LET a$=INKEY$:IF a$<>"x" THEN GO SUB
   3000:REM leer tecla
40 LET m=10:GO SUB 2000:REM impulso
50 IF a$<>"x" THEN GO TO 30
60 OUT 31,0
70 STOP
80 :
1000 REM **** inicializar ****
1010 LET fw=4:LET bw=2:LET lf=6:LET rt=0
1020 LET dr=fw
1030 RETURN
1040 :
2000 REM **** impulso ****
2010 FOR c=1 TO m
2020 OUT 31,dr+9
2030 OUT 31,dr+1
2040 NEXT c
2050 RETURN
2060 :
3000 REM **** leer teclas ****
3010 IF a$="t" THEN LET dr=fw:RETURN
3020 IF a$="b" THEN LET dr=bw:RETURN
3030 IF a$="f" THEN LET dr=lf:RETURN
3040 IF a$="h" THEN LET dr=rt:RETURN
3050 RETURN
```

## Parachoques Spectrum

```
10 REM **** parachoques spectrum ****
15 CLEAR 32499:LET st=32500
20 GO SUB 500:REM inicializar
25 GO SUB 3000: REM cargar lenguaje maquina
30 LET dr=fw
40 REM **** impulso adelante ****
50 LET cc=0
60 GO SUB 1000:LET cc=cc+1:REM impulso
70 REM ** comprobar parachoques **
80 LET nm=192:GO SUB 2000: REM efectuar AND
90 IF USR st=192 THEN GO TO 60
100 REM **** vuelta al comienzo ****
110 LET dr=bw
120 FOR i=1 TO cc
130 GO SUB 1000:REM impulso
140 NEXT i
150 OUT 31,0:STOP
500 REM **** inicializar ****
510 LET fw=4:LET bw=2:LET lf=6:LET rt=0
520 RETURN
1000 REM **** impulso ****
1010 OUT 31,dr+9
1020 OUT 31,dr+1
1030 RETURN
2000 REM **** efectuar AND ****
2010 POKE st+1,IN 31
2020 POKE st+3,nm:RETURN
3000 REM **** cargador lenguaje maquina ****
3010 FOR i=st TO st+8
3020 READ a:POKE i,a
3030 NEXT i
3040 DATA 62,0,14,0,161,6,0,79,201
3050 RETURN
```

## Complementos al BASIC

En el programa Calibrado Lineal suprima las líneas 20, 30 y 70 e introduzca los siguientes cambios:

```
80 LET dr=fw
100 LET a$=INKEY$:IF a$=""THEN GO TO 100
280 OUT 31,dr+9
290 OUT 31,dr+1
```

En el programa Calibrado Angular suprima las líneas 20, 30 y 50 e introduzca las siguientes modificaciones:

```
60 LET dr=rt
80 LET dr=fw
100 OUT 31,dr+9
110 OUT 31,dr+1
```

En el programa Medición Robot suprima las líneas 1010, 1020, 1030, 7010 y 7510 e introduzca las siguientes modificaciones:

```
2050 GO SUB 8100:IF USR st<>lb THEN GO TO
     2040
2090 GO SUB 8100:IF USR st<>lb THEN GO TO
     2080
2150 GO SUB 8100:IF USR st<>rb THEN GO TO
     2140
2190 GO SUB 8100:IF USR st<>rb THEN GO TO
     2180
2200 OUT 31,0
3010 CLS
3520 GO SUB 8100:IF USR st=nb THEN GO TO
     3510
4010 GO SUB 8100:IF USR st=rb THEN LET
     ss=rt: GO SUB 5000:RETURN
4020 GO SUB 8100:IF USR st=lb THEN LET
     ss=lf: GO SUB 5000:RETURN
5020 GO SUB 8100:IF USR st=nb THEN GO TO
     5010
6080 GO SUB 8100:IF USR st=nb THEN GO TO
     6070
8100 REM **** efectuar AND ****
8110 POKE st+1,192:POKE st+3,IN 31:RETURN
8200 REM **** cargador lenguaje maquina ****
8210 FOR i=st TO st+8
8220 READ a:POKE i,a
8230 NEXT i
8240 DATA 62,0,14,0,161,6,0,79,201
8250 RETURN
```

# Viejos y nuevos

**Se inicia aquí un estudio sobre las relaciones entre el BASIC y el sistema operativo en el BBC Micro**

La ROM para el BASIC en el BBC Micro ocupa la zona de memoria situada entre las direcciones &8000 y &BFFF. Esta zona se conoce también como espacio paginado de la ROM. Quiere decir que, mediante un sabio empleo del direccionamiento paginado, pueden corresidir en la máquina varios chips de ROM y emplear el mismo bloque de memoria. Sólo una de estas ROM está en activo, o sea "paginada", en cada ocasión. Son ejemplos de ROM que ocupan esta área de la memoria el chip DFS, el Econet, utilidades del tipo Wordwise, View o Disk Doctor, y otros lenguajes como FORTH y BCPL. La ventaja que reporta este empleo de ROM paginadas es obvia: ahorra memoria.

El que esté en activo una u otra ROM paginada depende del valor de *registro paginador (paging register)*, que forma parte del hardware controlado por el OS. Es posible acceder a un máximo de 16 ROM paginadas al mismo tiempo. El OS examina estas ROM cuando sucede algo inesperado. Por ejemplo, si una instrucción * no es reconocida por el OS, éste interroga a las ROM presentes en la máquina antes de dar el mensaje BAD COMMAND. Por ejemplo, *INFO no será recibida en la ROM del OS y será pasada a alguna de las ROM existentes en la máquina. Si está colocada, la ROM del DFS la aceptará y actuará en consonancia.

## Versiones del BASIC del BBC

Dos versiones del BASIC han sido diseñadas para los micros BBC empleados en Gran Bretaña. Son el BASIC I y el BASIC II. Basta con pulsar CTRL-Break, digitar REPORT y pulsar finalmente Return para saber qué versión es la que alberga un micro determinado. Si se imprime un mensaje de *copyright* fechado en 1982, la versión es BASIC II. Si la fecha es 1981, se trata de la versión más antigua, el BASIC I. La nueva versión eliminó algunos de los gazapos que se deslizaron en la vieja y añadió otras características.

Estas nuevas características incluyen la instrucción de archivo, OPENUP, y cambian el papel desempeñado por OPENIN. En el BASIC II, OPENIN abrirá un fichero sólo para entrada; en el BASIC I la apertura de un fichero puede servir, con esta instrucción, tanto para entrada como para salida, lo que constituye una característica valiosa para los ficheros en disco de acceso directo. OPENUP en el BASIC II abre un fichero tanto para entrada como para salida, y puede considerarse el equivalente del antiguo OPENIN del BASIC I. Este cambio produce las complicaciones propias de las transferencias de programas entre máquinas. Si usted escribió un programa que incluía instrucciones OPENIN en BASIC II, a la hora de cargarlo en una máquina con BASIC I se verá en un buen lío. El símbolo *(token)* que el BASIC II emplea para representar OPENIN no es reconocido por el BASIC I. Por ejemplo, esta sentencia:

```
Y%=OPENIN("RAUL")
```

en BASIC II, se lee

```
Y%=("RAUL")
```

en BASIC I, dando el consiguiente mensaje de error.

Las restantes facilidades del BASIC II tienen un interés especial para los programadores en lenguaje máquina, y en particular hay que mencionar una mejora importante. En el BASIC I si se requería la incorporación de datos a los programas en código máquina, se tenía que abandonar el ensamblador del BBC y usar los operadores ?, ! y $ para guardar los datos en la memoria.

Por ejemplo:

```
1000 LDA dato
1010 .dato
1020 ]
1030 ?P%=00
1040 P%=P%+1
1050 OPT pass
```

Lo que resulta algo complicado, y muchos ensambladores nos ofrecen un camino que lo evita por medio de *directivas de ensamblador*. Se trata de instrucciones que indican al ensamblador la tarea concreta que ha de realizar: en nuestro programa OPT es una directiva de ensamblador. Pero el BASIC II proporciona cuatro directivas de ensamblador más: EQUB guarda un byte particular en la memoria; EQUD guarda dos; EQUW cuatro y EQUS permite almacenar una serie en la memoria. El programa anterior puede escribirse así, en BASIC II:

```
1000 LDA dato
1010 ,dato EQUB 00
1020 ...resto del programa
```

Esto hará que el registro A se cargue con el byte contenido en la dirección dato, que en este ejemplo es cero.

Otro ejemplo sería:

```
1000 .mensaje EQUS "Hola chicos"
```

En este caso, los bytes que representan el mensaje Hola chicos se colocarán en la memoria en la dirección etiquetada .mensaje. Cuando se emplean todas estas directivas de ensamblador, P% se actualiza automáticamente teniendo en cuenta los datos.

Otro añadido útil del BASIC II es la instrucción OSCLI. Ya comentamos cómo se pasan las instrucciones * al OS del BBC mediante la llamada OSCLI en la dirección &FFF7. Con el BASIC I había que establecer las variables X% e Y% (o los registros X e Y) con la dirección en memoria de la serie que representaba la instrucción que había que pasar. La

nueva instrucción OSCLI en el BASIC II hace mucho más fácil la tarea. Comparémoslas:

| BASIC I | BASIC II |
|---|---|
| $&A00="*TAPE"<br>X%=&A00 MOD 256<br>Y%=&A00 DIV 256<br>CALL &FFF7 | OSCLI ("*TAPE") |

Además de estas dos versiones del BASIC, existe un BASIC II modificado para el mercado estadounidense; las alteraciones de esta versión sólo hacen referencia a las características de la visualización. También el segundo procesador 6502 contiene su propia versión del BASIC II llamada HI BASIC. No parece sino una mezcla de las versiones estadounidense y del BASIC II propiamente dicho. Desde el punto de vista del programador es una versión idéntica al BASIC II. Finalmente, añadiremos que ambas versiones del BASIC emplean del mismo modo la memoria del BBC Micro.

## Espacio de trabajo para el lenguaje

El área de memoria comprendida entre &400 y &7FF se conoce como *espacio de trabajo para el lenguaje (language workspace)*; se reserva al lenguaje que actualmente se está usando. Además, la página cero entre las direcciones &00 y &6F queda reservada para ese mismo lenguaje en uso, aunque parte de dicho espacio, el comprendido entre &50 y &6F no parece que sea muy utilizado. La página 4 de la memoria la emplea el BASIC para el almacenamiento de las variables; de &400 a &46B es el área de memoria donde los valores asignados a las variables enteras residentes quedan almacenados (variables tales como A% o X%). La variable @% se almacena en los cuatro bytes que van del &400 al &403 (con el byte menos significativo en &400), mientras que A% se almacena desde &404 a &407, y así sucesivamente hasta la última variable, que es Z%.

El área de memoria entre &480 y &4F9 sirve de índice para el intérprete BASIC; apunta a los lugares de la memoria donde se almacenan otras variables enteras o de coma flotante, matrices o series. Por ejemplo, los dos bytes de &482 y &483 apuntan a un área de la memoria (entre TOP y HIMEM) donde pueden encontrarse detalles de todas las variables actualmente empleadas cuyos nombres comiencen por la letra A. La dirección guardada en las posiciones &4F6 y &4F7 se refiere al almacenamiento de los procedimientos del BASIC, y la dirección guardada en las posiciones &4F8 y &4F9, al almacenamiento de las funciones.

El área de memoria entre &500 y &5FF sirve de pila para el intérprete del BASIC. Lo que no ha de confundirse con la pila del 6502 almacenada en la página 1 de la memoria. La página 5 la emplea el BASIC para almacenar los detalles de RETURN para toda llamada GOSUB, y la información necesaria para la ejecución de los ciclos FOR...NEXT y REPEAT...UNTIL.

La página 6 de la memoria sirve al BASIC o bien para el tratamiento de variables de serie o bien para el caso que se emplee una instrucción CALL que permita ejecutar un programa en código máquina, de lo que se hablará más adelante. El BBC BASIC utiliza la página 7 como buffer de entrada para cuando se digitan instrucciones BASIC o líneas de programa desde el teclado. Todo lo que se digita se sitúa en esta área de la memoria en espera de ser procesado. Si es una instrucción directa, se interpreta y se ejecuta; si es una línea de programa se coloca en la posición correspondiente dentro del programa. Concluimos así este repaso detallado del uso que el BASIC hace del espacio de trabajo para el lenguaje en un BBC Micro. Algo que resulta de utilidad en esta área de memoria es que si se deja de emplear el BASIC, queda disponible para cualquier otro uso.

## Tratamiento de errores en BASIC

Quién más quién menos todos cometemos errores en nuestros programas en BASIC, y hemos experimentado cómo nos responde el ordenador con sus mensajes de error. Lo que es bastante interesante en este tema del tratamiento de los errores en el BBC Micro es que podemos aprovechar el "manejador de errores" para elaborar nuestras propias sentencias de error, completándolas con mensajes y números. Esto es muy útil cuando conjuntamos programas en código máquina con otros en BASIC. La rutina encargada de tratar los errores generados por un programa BASIC tiene su dirección en el contenido del vector llamado BRKV, el cual se sitúa en la dirección &202 y &203. Veamos cómo se da entrada a la rutina a la cual apunta este vector.

Cuando el BASIC tiene que generar un error porque ha ocurrido algo ilegal (p. ej., dividir por cero), éste ejecuta una instrucción del 6502 en lenguaje máquina llamada BRK. Lo que produce un salto a la rutina especificada en la dirección contenida en BRKV (*break vector*: vector de ruptura). En condiciones normales esta rutina espera encontrar la instrucción BRK seguida de una serie de bytes, y establecida de la siguiente manera:

```
BRK
Número del error (un byte)
Mensaje de error (serie de bytes)
Fin del mensaje (un byte cero)
```

Por ejemplo, para el error número 4, que genera el mensaje *Mistake* (equivocado), se ejecutaría la siguiente rutina para hacer entrar al manejador de errores.

| Byte | Comentario |
|---|---|
| BRK | Instrucción |
| 4 | Número de error |
| M | |
| i | |
| s | Código ASCII |
| t | del mensaje |
| a | de error |
| k | |
| e | |
| 00 | Fin del mensaje |

El manejador de errores del BASIC establece entonces la variable de sistema ERL para que guarde el

número de línea en que ocurrió el error. Seguidamente se coloca el número del error en ERR (en nuestro caso 4) y el manejador se encarga de que a continuación se evalúe REPORT y que se imprima el mensaje *Mistake*.

La instrucción BRK se puede incluir en los programas en código máquina, por lo que podemos elaborar nuestros propios mensajes de error. Cualquier mensaje de error que generemos se aceptará en BASIC mediante la instrucción ON ERROR GOTO, como si el error lo generara el intérprete del BASIC. Por ejemplo, el siguiente fragmento de un programa en lenguaje máquina proporciona el mensaje de error *Number too big* (número demasiado grande):

```
1010 BRK
1020 EQUB 254
1030 EQUS "Number too big"
1040 EQUB 00
```

Una vez ejecutado, si imprimimos ERR nos dará 254 y REPORT dará *Number too big*.

Es posible alterar el contenido de este vector y cambiar el modo en que el BBC Micro responde a los errores. Empleando este vector se pueden añadir nuevas instrucciones al BASIC del BBC, pero esto requiere una técnica de programación algo compleja.

En el próximo capítulo de este curso examinaremos la forma en que se produce la interacción entre el BASIC y el sistema operativo del BBC Micro.

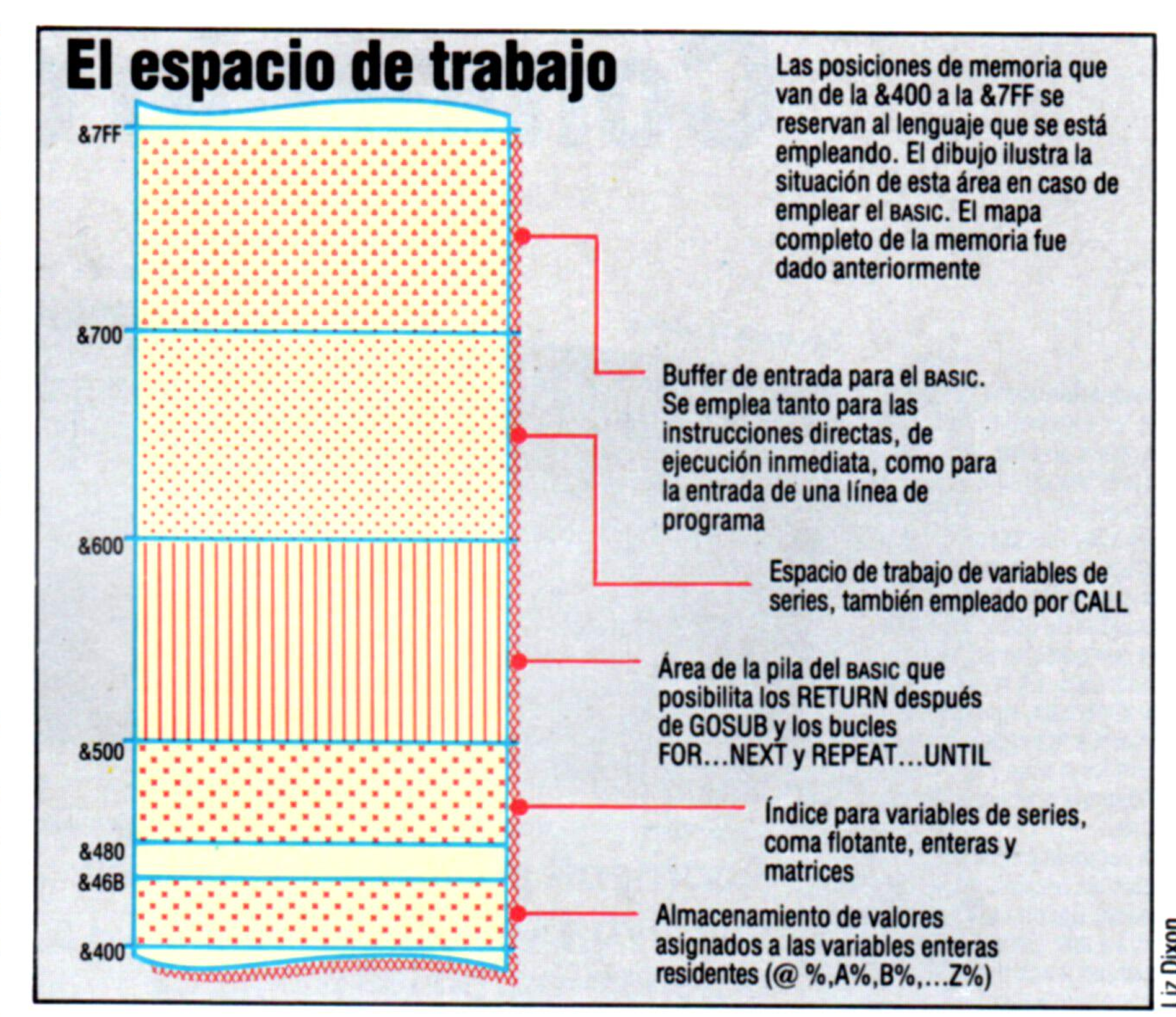
## Repaso de rutinas

Vamos a dar una lista de las direcciones en la ROM de algunas de las rutinas en BASIC del BBC Micro que hasta ahora se han mencionado. El examen de dichas rutinas le hará adquirir mayor conciencia del modo de operar del BASIC. No todas las aquí reseñadas terminan con RTS, por tanto no es recomendable su llamada indiscriminadamente desde programas propios, pero el examen de la técnica empleada puede ayudarnos a mejorar la propia habilidad de programación.

| **Punto de entrada a la ROM** | **&8000** |
|---|---|
| Sentencia BGET | &BF6F |
| Sentencia BPUT | &BF58 |
| Sentencia CHAIN | &BF2A |
| Sentencia CLOSE | &BF99 |
| Función EOF | &ACB8 |
| Función INKEY | &ACAD |
| Función INKEY$ | &B026 |
| Sentencia INPUT | &BA44 |
| Sentencia INPUT# | &B9CF |
| Instrucción LOAD | &BF24 |
| Función OPENIN | &BF78 |
| Función OPENOUT | &BF7C |
| Función OPENUP | &BF80 |
| Sentencia PRINT# | &8D2B |
| Instrucción SAVE | &BEF3 |
| Sentencia TIME | &92C9 |
| Función TIME | &AEB4 |
| Sentencia VDU | &942F |

La información es por cortesía de Acorn Computers. 

## El espacio de trabajo




Las posiciones de memoria que van de la &400 a la &7FF se reservan al lenguaje que se está empleando. El dibujo ilustra la situación de esta área en caso de emplear el BASIC. El mapa completo de la memoria fue dado anteriormente

Liz Dixon

## Variantes del BASIC I y BASIC II en el BBC

| Instrucción o error | BASIC I | BASIC II |
|---|---|---|
| REPORT | Da el mensaje 1981 | Da el mensaje 1982 |
| OPENUP | No implementada | Abre un fichero para lectura y escritura (como OPENIN en el BASIC I) |
| OPENIN | Abre fichero para lect. y escr. | Abre fichero para lectura solamente |
| OPT n | OPT 1 hasta 3 son las únicas implementadas | Se conocen OPT 1 hasta 7. Las cuatro nuevas instrucciones equivalen a las otras tres, pero el código se ensambla a la dirección contenida en la variable 0% |
| 0% | Sin uso práctico | Función diseñada en una entrada previa (instr. OPTn) |
| EQUB, EQUD, EQUS, EQUW | No implementadas | Permite al programador incluir datos en programas en código máquina sin abandonar el ensamblador (como vimos) |
| INSTR (a$,b$) | Grave error: se producía un crac cada vez que a$ era más corta que b$ | Corregido |
| Núms. reales | Almacenados con precisión de 9 cifras | Almacenados con precisión de 10 cifras |
| OSCLI | No implementada | Posibilita el paso de instrucciones al OS desde variables de series del BASIC |

# Comercio cósmico

Comienza la misión




Diagrama de corto alcance




Estación espacial




**Comportamiento "elitista"**
En las fotografías vemos tres escenas de *Elite*, incluyendo la pantalla de apertura, que representa a la nave Cobra Mark III. Completar el juego puede llevar meses y, por consiguiente, posee la facilidad de guardarlo (SAVE) en cualquier punto de su desarrollo. En la parte inferior de la pantalla aparecen los diversos indicadores necesarios durante el viaje y un mapa tridimensional por radar de la zona.
La segunda pantalla muestra las estrellas próximas a su posición actual. Desplazando el punto de mira a una estrella escogida, aparecerá una pantalla que proporciona datos acerca del planeta: su forma de gobierno, tipo de economía, etc. La tercera pantalla muestra la estación espacial. Para realizar el acoplamiento el jugador debe conducir su nave hasta un pequeño puerto rectangular

## Acornsoft promete premios en efectivo y certificados para quienes logren las mayores puntuaciones en su juego "Elite"

*Elite*, de Acornsoft, combina elementos de los juegos recreativos, de aventuras y de estrategia. El objetivo del juego consiste en pasar a formar parte de la Élite. Para conseguirlo, el jugador se debe convertir en un mercader con éxito entre los miles de planetas de ocho galaxias.

Para permanecer con vida el tiempo suficiente para amasar fortuna y unirse a la Élite, el jugador no sólo debe convertirse en un hábil navegante y guerrero, sino que también necesitará buenas aptitudes para los negocios, cierta dosis de astucia y ser capaz de moverse por la difusa línea que separa lo legal de lo ilegal. Se pueden obtener enseguida enormes beneficios haciendo contrabando entre los planetas o convirtiéndose en pirata, pero el jugador ha de tener cuidado. Es posible que la policía se entere de sus actividades y podría verse catalogado como delincuente o fugitivo... y las veloces naves de la policía disparan a matar.

El juego comienza con el jugador acoplado en una estación espacial en la superficie del planeta Lave. Su nave es una Cobra Mark III; está equipada con un láser de impulso frontal y posee una capacidad de carga de 20 toneladas. Antes de partir, es aconsejable familiarizarse con los muchos controles y opciones disponibles, así como practicar las maniobras de acoplamiento ya que, de lo contrario, uno puede estrellarse en su primera misión. Si usted pierde demasiado tiempo intentando acoplarse a una estación espacial u orbitando alrededor de un planeta hostil, seguramente atraerá la atención de los piratas.

Una vez realizado el acoplamiento, llega la hora de hacer negocios. Comparando los precios del planeta con las mercancías que usted posee, debe decidir si le conviene más vender o conservarlas hasta hallar condiciones más favorables en algún otro sitio. Uno de los pocos inconvenientes del juego, al igual que ocurre en la mayoría de los otros juegos de comercio existentes para ordenadores, es que no hay ninguna opción para el regateo. El jugador debe aceptar el precio ofrecido o bien rechazarlo.

El verdadero secreto del comercio consiste en decidir qué productos llevar a cada planeta. Por ejemplo, se puede comprar alimentos a bajo precio en un planeta agrícola, y venderlos a buen precio en un planeta industrial. Luego se puede llevar maquinaria al país agrícola. El jugador debe también tener en cuenta la estructura política del planeta y el tipo de seres que lo habitan. Una estructura política caótica significa que es probable que haya piratas, y algunas especies de alienígenas tienen más tendencia a estafarlo o eliminarlo que otras.

Los gráficos tridimensionales de *Elite* son excelentes. Los planetas, las estaciones y las naves espaciales están dibujadas de forma esquemática y acostumbrarse a ellas cuesta un poco al principio, pero uno comienza enseguida a apreciar la ventaja de una resolución más alta y la capacidad de girar en las tres dimensiones. Esto se aprovecha al máximo durante los acoplamientos y los combates. Las estaciones espaciales sólo poseen una única entrada y la aproximación se debe realizar en ángulo recto. Cuando uno entra en combate, la capacidad de ver la nave enemiga (de la cual hay muchos tipos diferentes) le añade al juego un gran realismo.

Además de la cassette o el disco se suministra un exhaustivo manual que incluye detalles relativos a todos los aspectos del juego, una historia basada en el juego llamada *The dark wheel* (La rueda oscura), una ficha que contiene un resumen de los controles, una plantilla para las teclas de función y un *poster* que ilustra las diversas clases de naves espaciales con la que es probable que se encuentre.

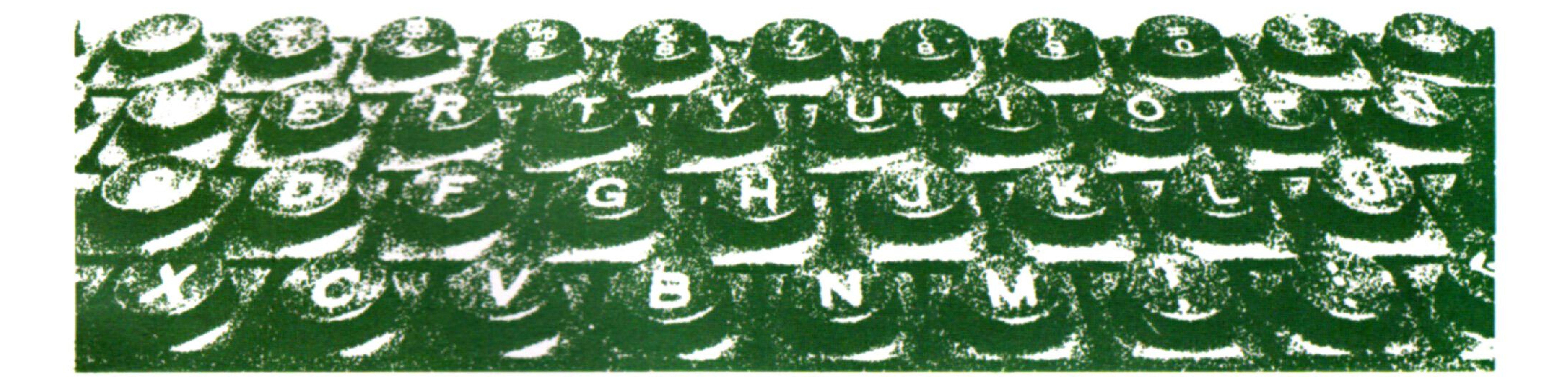
**Elite:** Para el BBC Micro y el Electron (palancas de mando opcionales)

**Editado por:** Acornsoft Ltd, Betjemin House, 104 Hills Road, Cambridge, CB2 1LQ, Gran Bretaña

**Autores:** Ian Bell, David Braben

**Formato:** Cassette o disco

Editorial Delta, S.A.



10076
9 788485 822836